# 자율주행자동차 제작

## 1. 아두이노 모터 조작 코드

```
String Speed;
char  LorR;
int  i, s;

int motorA1 = 3;
int motorA2 = 11;
int motorB1 = 5;
int motorB2 = 6;

unsigned int motorA_SPEED = 0, motorB_SPEED = 0;
boolean motorA_DIR = 0, motorB_DIR = 0;

byte DataToRead[6];

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(motorA1, OUTPUT);
  pinMode(motorA2, OUTPUT);
  pinMode(motorB1, OUTPUT);
  pinMode(motorB2, OUTPUT);
}

void loop() {
  DataToRead[5] = '\n';
  Serial.readBytesUntil(char(13), DataToRead, 5);

/* For Debugging, send string to RPi */
  for (i = 0; i < 6; i++) {
    Serial.write(DataToRead[i]);
    if (DataToRead[i] == '\n') break;
  }
/* End of Debugging */

  LorR = DataToRead[0];
  Speed = "";
  for (i = 1; (DataToRead[i] != '\n') && (i < 6); i++) {
    Speed += DataToRead[i];
  }
  s = Speed.toInt();

  if (LorR == 'L') {
    // Turn left wheel with speed s
    if (s < 0) {
      motorA_DIR = HIGH;
    }
    else {
      motorA_DIR = LOW;
```

```
    }
    motorA_SPEED = abs(s);
    digitalWrite(motorA1, motorA_DIR);
    analogWrite(motorA2, motorA_SPEED);
  }

  else if (LorR == 'R') {
    // Turn right wheel with speed s
    if (s < 0) {
      motorB_DIR = HIGH;
    }
    else {
      motorB_DIR = LOW;
    }
    motorB_SPEED = abs(s);
    analogWrite(motorB1, motorB_SPEED);
    digitalWrite(motorB2, motorB_DIR);
  }

  else if (LorR == 'S') {
    // Stop Motors
    digitalWrite(motorA1, LOW);
    analogWrite(motorA2, 0);
    analogWrite(motorB1, 0);
    digitalWrite(motorB2, LOW);

    delay(500);
  }
}
```

Code 1 Serial_edit.ino

## 2. 딥러닝을 이용한 방법

### 2.1. 1차 시도

#### 2.1.1. 이미지 라벨링

labelImg 툴을 이용하였으며, 그 결과를 xml 파일로 저장하였다. 트랙의 꺾인 정도에 따라 {-1}, {0}, {1}으로 분류하여 라벨링을 진행하였다. 그리고 이를 강의록 게시판에 제공된 예시 데이터와 병합하였다. 이미지는 정방형으로 아랫부분을 자른 뒤, 16*16 크기로 조정하고 흑백으로 변환 및 이진화 한 후 색상을 반전시켰다.

```
- <source>
    <database>Unknown</database>
  </source>
- <size>
    <width>1080</width>
    <height>1920</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
- <object>
    <name>-1</name>
    <pose>Unspecified</pose>
    <truncated>1</truncated>
    <difficult>0</difficult>
  - <bndbox>
      <xmin>1</xmin>
      <ymin>1050</ymin>
      <xmax>1080</xmax>
      <ymax>1920</ymax>
    </bndbox>
  </object>
```

```
- <source>
    <database>Unknown</database>
  </source>
- <size>
    <width>1080</width>
    <height>1920</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
- <object>
    <name>0</name>
    <pose>Unspecified</pose>
    <truncated>1</truncated>
    <difficult>0</difficult>
  - <bndbox>
      <xmin>3</xmin>
      <ymin>994</ymin>
      <xmax>1080</xmax>
      <ymax>1920</ymax>
    </bndbox>
  </object>
```

```
- <source>
    <database>Unknown</database>
  </source>
- <size>
    <width>1080</width>
    <height>1920</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
- <object>
    <name>-1</name>
    <pose>Unspecified</pose>
    <truncated>1</truncated>
    <difficult>0</difficult>
  - <bndbox>
      <xmin>1</xmin>
      <ymin>1050</ymin>
      <xmax>1080</xmax>
      <ymax>1920</ymax>
    </bndbox>
  </object>
```

Figure 1 {-1}, {0}, {-1} 그룹

```
dirs_dest = os.getcwd() + '/img_converted/'
img_converted = []
for img_name in img_original:
    img = cv2.imread(dirs+img_name, cv2.IMREAD_GRAYSCALE)
    # 아랫부분을 중심으로 1080*1080으로 자르기
    crop_img = img[840:, :]
    # 16*16으로 줄이기
    resize_img = cv2.resize(crop_img, (16,16), interpolation=cv2.INTER_AREA)
    # 색상 반전
    for x in range(16):
        for y in range(16):
            if resize_img[y, x] < 128:
                resize_img[y, x] = 255
            else:
                resize_img[y, x] = 0
    # local에 저장
    cv2.imwrite(dirs_dest+img_name, resize_img)
    # list에 저장
    img_converted.append(resize_img)
```

Code 2 이미지 변환에 사용한 Python 코드

#### 2.1.2. 학습 결과

loss를 줄일 수 있었던 조건의 조합은 다음과 같다. 하지만, prediction 결과에 오류가 많았다. 따라서 라벨링을 더 세밀하게 진행하고 해당 데이터만을 사용하기로 결정하였다.

```
trainingdata_all.pickle, test_size = 0.33, random_state = 123, epoch = 2
```

```
- Epoch 1/2
    - 4920/4920 [==============================] - 7s 1ms/step - loss: 38.6281
- Epoch 2/2
    - 4920/4920 [==============================] - 7s 1ms/step - loss: 0.1972
```

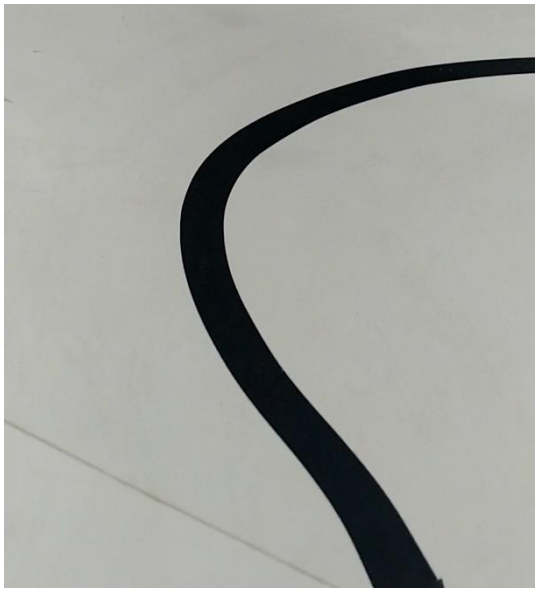
```
trainingdata_all.pickle, test_size = 0.33, random_state = 321, epoch = 2
```

```
- Epoch 1/2
    - 4920/4920 [==============================] - 7s 1ms/step - loss: 68.4014
- Epoch 2/2
    - 4920/4920 [==============================] - 7s 1ms/step - loss: 0.1841
```

### 2.2. 2차 시도

#### 2.2.1. 이미지 라벨링

labelImg 툴을 이용하였으며, 그 결과를 xml 파일로 저장하였다. 트랙의 꺾인 정도에 따라 {-1}, {-0.5}, {0}, {0.5}, {1}으로 분류하여 총 1,131장의 이미지에 대해 라벨링을 진행하였다.



```
- <source>
    <database>Unknown</database>
  </source>
- <size>
    <width>1080</width>
    <height>1920</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
- <object>
    <name>-0.5</name>
    <pose>Unspecified</pose>
    <truncated>1</truncated>
    <difficult>0</difficult>
  - <bndbox>
      <xmin>1</xmin>
      <ymin>1050</ymin>
      <xmax>1080</xmax>
      <ymax>1920</ymax>
    </bndbox>
  </object>
```

```
- <source>
    <database>Unknown</database>
  </source>
- <size>
    <width>1080</width>
    <height>1920</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
- <object>
    <name>0.5</name>
    <pose>Unspecified</pose>
    <truncated>1</truncated>
    <difficult>0</difficult>
  - <bndbox>
      <xmin>3</xmin>
      <ymin>1054</ymin>
      <xmax>1080</xmax>
      <ymax>1920</ymax>
    </bndbox>
  </object>
```

Figure 2 추가된 {-0.5}, {0.5} 그룹

### 2.2.2. 학습 결과

loss를 줄일 수 있었던 조건의 조합은 다음과 같다. loss가 크게 감소한만큼 prediction 결과 또한 오류가 크게 줄었으나, 실제로 RC카를 움직이면 그 경로가 트랙의 모양과 일치하지 않고 불규칙했다. 따라서 OpenCV를 이용한 방법을 택하였다.

```
trainingdata_add2.pickle, test_size = 0.33, random_state = 321, epoch 6
- Epoch 1/6
    - 757/757 [==============================] - 1s 1ms/step - loss: 666.0933
- Epoch 2/6
    - 757/757 [==============================] - 1s 1ms/step - loss: 1.5435
- Epoch 3/6
    - 757/757 [==============================] - 1s 1ms/step - loss: 0.1681
- Epoch 4/6
    - 757/757 [==============================] - 1s 1ms/step - loss: 0.1258A:
- Epoch 5/6
```

```
    - 757/757 [==============================] - 1s 2ms/step - loss: 0.0964
 - Epoch 6/6
    - 757/757 [==============================] - 1s 1ms/step - loss: 0.0687
```

## 3. OpenCV를 이용한 방법

### 3.1. Utils.py (강의록 게시판에 제공되고 있는 코드이다.)

```
# -*- coding: utf-8 -*-
import numpy as np
import cv2
import time
from Image import *

# 그림을 slices 의 수만큼 조각낸다
def SlicePart(im, images, slices):
    height, width = im.shape[:2]
    sl = int(height/slices);
    points = []

    for i in range(slices):
        part = sl*i
        crop_img = im[part:part+sl, 0:width]
        #조각난 이미지 crop_img를 images[]에 저장
        images[i].image = crop_img
        #Image.py에서 윤곽선을 그리고 무게중심을 표시
        cPoint = images[i].Process()
        points.append(cPoint)
    return points

#조각난 이미지를 다시 합친다
def RepackImages(images):
    img = images[0].image
    for i in range(len(images)):
        if i == 0:
            img = np.concatenate((img, images[1].image), axis=0)
        if i > 1:
            img = np.concatenate((img, images[i].image), axis=0)

    return img

def Center(moments):
    if moments["m00"] == 0:
        return 0

    x = int(moments["m10"]/moments["m00"])
    y = int(moments["m01"]/moments["m00"])

    return x, y
```

```
def RemoveBackground(image, b):
    up = 50 #100
    # create NumPy arrays from the boundaries
    lower = np.array([0, 0, 0], dtype = "uint8")
    upper = np.array([up, up, up], dtype = "uint8")
    #----------------COLOR SELECTION-------------- (Remove any area that is whiter
than 'upper')
    if b == True:
        mask = cv2.inRange(image, lower, upper)
        image = cv2.bitwise_and(image, image, mask = mask)
        image = cv2.bitwise_not(image, image, mask = mask)
        image = (255-image)
        return image
    else:
        return image
    #////////////////COLOR SELECTION/////////////
```

Code 3 Utils.py

3.2. Image.py (강의록 게시판에 제공되고 있는 코드에서 일부를 수정하였다.)

```
# -*- coding: utf-8 -*-
import numpy as np
import cv2

class Image:

    def __init__(self):
        self.image = None
        self.contourCenterX = 0
        self.MainContour = None

    def Process(self):
    #이미지를 흑백으로 변환한 뒤 Threshold 값을 기준으로 0 또는 1로 값을 정한다
        imgray = cv2.cvtColor(self.image,cv2.COLOR_BGR2GRAY) #Convert to Gray Scale
        ret, thresh = cv2.threshold(imgray,50,255,cv2.THRESH_BINARY_INV) #Threshold 값을
100에서 50으로 조정하였다

        self.contours,                                          _                    =
cv2.findContours(thresh,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)[-2:] #Get contour

        self.prev_MC = self.MainContour
        if self.contours:
            self.MainContour = max(self.contours, key=cv2.contourArea)

            self.height, self.width  = self.image.shape[:2]

            self.middleX = int(self.width/2) #Get X coordenate of the middle point
            self.middleY = int(self.height/2) #Get Y coordenate of the middle point

            self.prev_cX = self.contourCenterX
            if self.getContourCenter(self.MainContour) != 0:
                self.contourCenterX = self.getContourCenter(self.MainContour)[0]
                if abs(self.prev_cX-self.contourCenterX) > 5:
                    self.correctMainContour(self.prev_cX)
            else:
```

```
            self.contourCenterX = 0

        self.dir       =              int((self.middleX-self.contourCenterX)       *
self.getContourExtent(self.MainContour))

        #윤곽선은 초록색, 무게중심은 흰색 원, 그림의 중앙 지점은 빨간 원으로 표시
        cv2.drawContours(self.image,self.MainContour,-1,(0,255,0),3)  #Draw  Contour
GREEN
        cv2.circle(self.image, (self.contourCenterX, self.middleY), 7, (255,255,255),
-1) #Draw dX circle WHITE
        cv2.circle(self.image, (self.middleX, self.middleY), 3, (0,0,255), -1) #Draw
middle circle RED

        font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(self.image,str(self.middleX-
self.contourCenterX),(self.contourCenterX+20,           self.middleY),           font,
1,(200,0,200),2,cv2.LINE_AA)

cv2.putText(self.image,"Weight:%.3f"%self.getContourExtent(self.MainContour),(self.con
tourCenterX+20, self.middleY+35), font, 0.5,(200,0,200),1,cv2.LINE_AA)
        return [self.contourCenterX, self.middleY]

    def getContourCenter(self, contour):
        M = cv2.moments(contour)

        if M["m00"] == 0:
            return 0

        x = int(M["m10"]/M["m00"])
        y = int(M["m01"]/M["m00"])

        return [x,y]

    def getContourExtent(self, contour):
        area = cv2.contourArea(contour)
        x,y,w,h = cv2.boundingRect(contour)
        rect_area = w*h
        if rect_area > 0:
            return (float(area)/rect_area)

    def Aprox(self, a, b, error):
        if abs(a - b) < error:
            return True
        else:
            return False

    def correctMainContour(self, prev_cx):
        if abs(prev_cx-self.contourCenterX) > 5:
            for i in range(len(self.contours)):
                if self.getContourCenter(self.contours[i]) != 0:
                    tmp_cx = self.getContourCenter(self.contours[i])[0]
                    if self.Aprox(tmp_cx, prev_cx, 5) == True:
                        self.MainContour = self.contours[i]
                        if self.getContourCenter(self.MainContour) != 0:
                            self.contourCenterX                                    =
self.getContourCenter(self.MainContour)[0]
```

Code 4 Image.py

3.3. control_motors.py (모터를 작동시키는 코드를 작성하였다.)

```
import time
import serial
ser = serial.Serial('/dev/serial/by-id/usb-Arduino_Srl_Arduino_Uno_7543134333435161
E1E1-if00',9600)


def L_Speed(speed):
    cmd_L = ("L%d\n" % speed).encode('ascii')
    print("My cmd is %s" % cmd_L)
    ser.write(cmd_L)

def R_Speed(speed):
    cmd_R = ("R%d\n" % speed).encode('ascii')
    print("My cmd is %s" % cmd_R)
    ser.write(cmd_R)

def MotorsStop():
    cmd = ("S%d\n" % 0).encode('ascii')
    print("My cmd is %s" % cmd)
    ser.write(cmd)
```

Code 5 control_motors.py

3.4. main.py (강의록 게시판에 제공되고 있는 코드에서 일부를 수정하였다.)

```
# -*- coding: utf-8 -*-
import socket
import sys
import os
import numpy as np
import pdb

import cv2
import time

from Image import *
from Utils import *
from control_motors import *

font = cv2.FONT_HERSHEY_SIMPLEX
direction = 0

#N_SLICES만큼 이미지를 조각내서 Images[] 배열에 담는다
Images=[]
N_SLICES = 3
middleX = 320

for q in range(N_SLICES):
    Images.append(Image())

# 카메라를 통해 이미지를 제공받는다
video_capture = cv2.VideoCapture(-1)
video_capture.set(3, 640)
```

```
video_capture.set(4, 240)

while(True):

    # Capture the frames
    ret, frame = video_capture.read()

    # 180도 회전 (카메라를 뒤집어서 설치하기 때문임)
    frame = cv2.rotate(frame, cv2.ROTATE_180)

    # Crop the image
    crop_img = frame#[60:120, 0:160]

    # Convert to grayscale
    gray = cv2.cvtColor(crop_img, cv2.COLOR_BGR2GRAY)

    # Gaussian blur
    blur = cv2.GaussianBlur(gray,(5,5),0)

    # Color thresholding
    ret,thresh1 = cv2.threshold(blur,50,255,cv2.THRESH_BINARY_INV) # 60

    cv2.imshow("1", thresh1)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break


    crop_img = RemoveBackground(crop_img, False)
    #cv2.imshow("2", crop_img)
    #if cv2.waitKey(1) & 0xFF == ord('q'):
    #    break
    if crop_img is not None:
        #이미지를 조각내서 윤곽선을 표시하게 무게중심 점을 얻는다
        Points = SlicePart(crop_img, Images, N_SLICES)
        print('Points : ', Points)

        ##### 모터 제어 코드 : (middleX - Points[2][0]) 값에 따라
        toFollow = middleX - Points[2][0]

        # (1) -20 ~ 20일 때 = 직진
        if toFollow <= 20 and toFollow >= -20:
            L_Speed(40)
            R_Speed(40)

        # (2) 음수일 때 = 우회전
        elif toFollow < 0:
            if toFollow >= -70:
                R_Speed(0)
                L_Speed(41)
            elif toFollow >= -120:
                R_Speed(0)
                L_Speed(42)
            elif toFollow >= -170:
                R_Speed(0)
                L_Speed(43)
            elif toFollow >= -220:
                R_Speed(0)
```

```
                L_Speed(44)
            elif toFollow >= -270:
                R_Speed(0)
                L_Speed(45)
            else:
                R_Speed(-20)
                L_Speed(40)

        # (3) 양수일 때 = 좌회전
        else:
            if toFollow <= 70:
                L_Speed(0)
                R_Speed(41)
            elif toFollow <= 120:
                L_Speed(0)
                R_Speed(42)
            elif toFollow <= 170:
                L_Speed(0)
                R_Speed(43)
            elif toFollow <= 220:
                L_Speed(0)
                R_Speed(44)
            elif toFollow <= 270:
                L_Speed(0)
                R_Speed(45)
            else:
                L_Speed(-20)
                R_Speed(40)
        #####

        #조각난 이미지를 한 개로 합친다
        fm = RepackImages(Images)

        #완성된 이미지를 표시한다
        cv2.imshow("Vision Race", fm)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        print('not even processed')
```

Code 6 main.py

### 3.5. main2.py (main.py에서 모터의 속도를 조정하였다.)

```
# -*- coding: utf-8 -*-
import socket
import sys
import os
import numpy as np
import pdb

import cv2
import time

from Image import *
from Utils import *
```

```
from control_motors import *

font = cv2.FONT_HERSHEY_SIMPLEX
direction = 0

#N_SLICES만큼 이미지를 조각내서 Images[] 배열에 담는다
Images=[]
N_SLICES = 3
middleX = 320

for q in range(N_SLICES):
    Images.append(Image())

# 카메라를 통해 이미지를 제공받는다
video_capture = cv2.VideoCapture(-1)
video_capture.set(3, 640)
video_capture.set(4, 240)

while(True):

    # Capture the frames
    ret, frame = video_capture.read()

    # 180도 회전 (카메라를 뒤집어서 설치하기 때문임)
    frame = cv2.rotate(frame, cv2.ROTATE_180)

    # Crop the image
    crop_img = frame#[60:120, 0:160]

    # Convert to grayscale
    gray = cv2.cvtColor(crop_img, cv2.COLOR_BGR2GRAY)

    # Gaussian blur
    blur = cv2.GaussianBlur(gray,(5,5),0)

    # Color thresholding
    ret,thresh1 = cv2.threshold(blur,50,255,cv2.THRESH_BINARY_INV) # 60

    cv2.imshow("1", thresh1)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break


    crop_img = RemoveBackground(crop_img, False)
    #cv2.imshow("2", crop_img)
    #if cv2.waitKey(1) & 0xFF == ord('q'):
    #    break
    if crop_img is not None:
        #이미지를 조각내서 윤곽선을 표시하게 무게중심 점을 얻는다
        Points = SlicePart(crop_img, Images, N_SLICES)
        print('Points : ', Points)

        ##### 모터 제어 코드 : (middleX - Points[2][0]) 값에 따라
        toFollow = middleX - Points[2][0]

        # (1) -20 ~ 20일 때 = 직진
```

```
        if toFollow <= 20 and toFollow >= -20:
            L_Speed(41)
            R_Speed(41)

        # (2) 음수일 때 = 우회전
        elif toFollow < 0:
            if toFollow >= -70:
                R_Speed(0)
                L_Speed(43)
            elif toFollow >= -120:
                R_Speed(0)
                L_Speed(45)
            elif toFollow >= -170:
                R_Speed(0)
                L_Speed(45)
            elif toFollow >= -220:
                R_Speed(0)
                L_Speed(46)
            elif toFollow >= -270:
                R_Speed(0)
                L_Speed(47)
            else:
                R_Speed(-20)
                L_Speed(40)

        # (3) 양수일 때 = 좌회전
        else:
            if toFollow <= 70:
                L_Speed(0)
                R_Speed(43)
            elif toFollow <= 120:
                L_Speed(0)
                R_Speed(45)
            elif toFollow <= 170:
                L_Speed(0)
                R_Speed(45)
            elif toFollow <= 220:
                L_Speed(0)
                R_Speed(46)
            elif toFollow <= 270:
                L_Speed(0)
                R_Speed(47)
            else:
                L_Speed(-20)
                R_Speed(40)
        #####

        #조각난 이미지를 한 개로 합친다
        fm = RepackImages(Images)

        #완성된 이미지를 표시한다
        cv2.imshow("Vision Race", fm)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        print('not even processed')
```

Code 7 main2.py