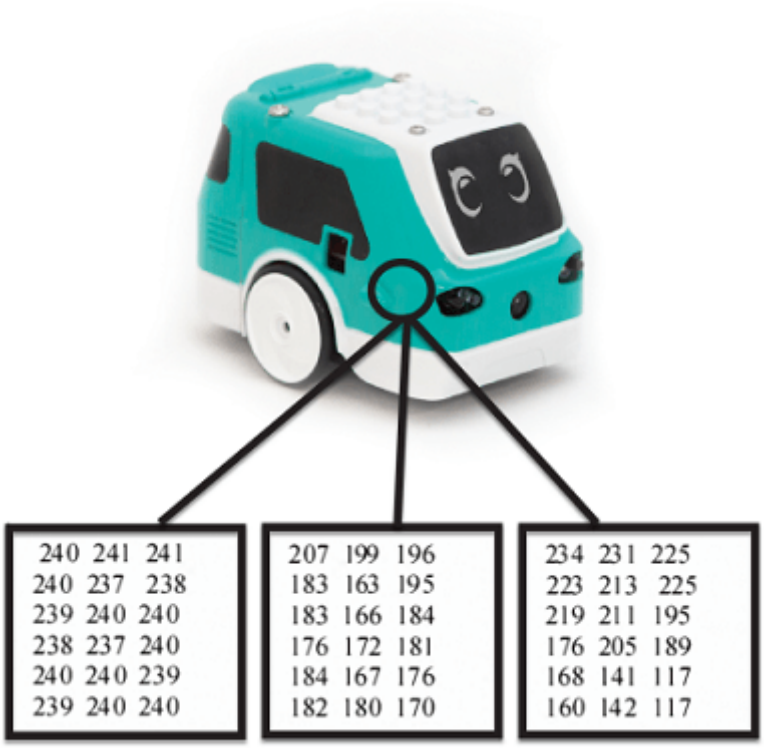
컴퓨터는 색상을 어떻게 보고 해석합니까?

주미가 색깔을 인식하는 훈련을 시작하기 전에, 주미가 색깔을 어떻게 보는지를 배울 필요가 있습니다.
그것은 우리가 색깔을 보는 것과는 매우 다릅니다! ## 이미지란? 이미지는 픽셀(사진 요소의 줄임말)이라고 불리는 작은 점들의 배열로 이루어져 있습니다. 픽셀은 하나의 색이나 색의 조합이 될 수 있습니다. 각각의 색상은 그 안에 얼마나 많은 빨강, 초록, 파랑의 색상이 있는지를 알려주는 세 개의 숫자로 표현됩니다. 이를 RGB 값이라고 하는데, 이는 빨간색, 녹색, 파란색을 의미합니다.

예를 들어, 청록색의 아름다운 그늘은 붉은색이 많지 않고, 녹색과 파란색이 많기 때문에 (27, 209, 197)과 비슷하게 보일 수 있습니다.
RGB의 각 값은 0에서 255 사이일 수 있기 때문에 각 색상마다 256개의 값을 선택할 수 있습니다. 그 결과 256^3, 즉 16,777,216개의 다른 색상 조합이 나오게 됩니다!

매트릭스란?

각각의 픽셀은 숫자로 표현될 수 있기 때문에 그림은 숫자로 된 격자 형태가 됩니다.
인간과 컴퓨터가 조금씩 다르게 이미지를 보기 시작하는 곳입니다.
인간은 우리가 다른 물체로 인식하는 색과 모양을 보지만, 컴퓨터는 행렬 혹은 매트릭스라고도 불리는 이러한 격자만을 봅니다.
각 숫자는 각 픽셀의 RGB 값을 나타냅니다.

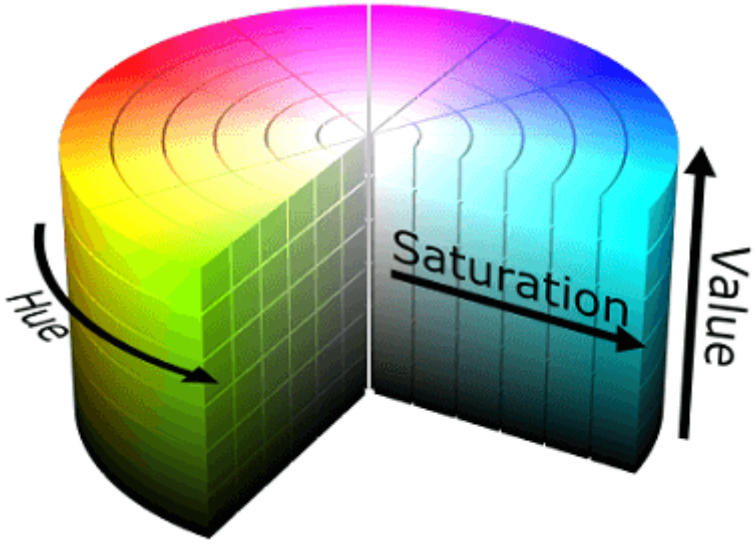


RGB 대신 HSV 사용

우리는 각 RGB 이미지를 HSV 색 공간으로 변환할 것입니다. HSV는 색도, 채도, 명도를 의미합니다.

- **색도(HUE)** 는 일반적으로 0-360 범위이며 색상을 나타냅니다 (여기에서는 0-180 범위).
- **채도(Saturation)** 는 색상의 선명도입니다.
- **명도(Value)** 는 색상이 얼마나 밝거나 어두운지를 나타냅니다.

컴퓨터 비전 응용 프로그램에서는 색상과 강도 값을 구분하므로 HSV 색 공간을 사용하는 것이 좋습니다.
이것은 그림자, 반사 및 기타 요인으로 인해 특정 색상이 매우 다르게 보일 수 있기 때문에 중요합니다.
HSV 색 공간은 보다 정확한 결과를 위해 이를 고려합니다.



주미에게 어떻게 색상을 가르칠까요?

머신러닝(기계학습)이란?

인간이 다양한 경험과 시행착오를 통해 지식을 배우는 것처럼, 컴퓨터에게 충분히 많은 데이터를 주고, 거기에서 일반적인 규칙(패턴)을 찾아내게 하는 방법을 말합니다. 이렇게 되면 컴퓨터는 입력하지 않은 정보에 대해서도 예측이나 결정을 할 수 있게 됩니다. 이것을 머신러닝이라고 합니다.

연습: 주미에게 세가지 색상 가르치기

데모의 색상 배우기로 이동하여 주미 상자에 있는 색상 카드 패킷에서 좋아하는 세 가지 색상을 선택하세요.
주미가 각 색상을 3D 좌표(색조, 채도 및 값)로 표시하는 방법에 주의하세요.

완료되면 기억하기 쉬운 프로젝트 이름으로 모델을 저장하는 것을 잊지 마세요.
다음 섹션에서 사용할 것이므로 모든 레이블 이름과 함께 적어 두세요.

이전 레슨에서 훈련한 모델을 사용해도 됩니다.

모델 로드 중

머신 러닝의 좋은 점은 모델을 저장하면 다시 사용할 수 있다는 것입니다!
이전 레슨에서 주미에게 가르칠 가장 좋아하는 세 가지 색상을 선택했습니다.
이제 파이썬 코드를 사용하여 모델을 로드하고 주미가 각 색상에 다르게 반응하도록 합니다!

라이브러리 가져오기

다음 라이브러리를 가져와서 운전, 카메라, 색상 분류에 필요한 기능에 접근하세요.

In [ ]:

```
from zumi.zumi import Zumi
from zumi.util.camera import Camera
from zumi.util.screen import Screen
from zumi.util.color_classifier import ColorClassifier
import time

camera = Camera()
screen = Screen()
zumi = Zumi()
```

모델 로드

모델을 로드하려면 ColorClassifier 라이브러리에서 함수를 호출합니다.
`load_model()`. 이 함수에는 프로젝트 이름인 매개변수가 필요합니다.
저장한 모델 이름을 기억하십니까?
아래 따옴표 안에 사용자 이름과 프로젝트 이름을 삽입하세요.
철자가 정확한지 확인하세요. 존재하지 않으면 오류가 발생합니다.

예를 들어 유져이름이 **Jun** 이고 저장한 이름이 **model1** 이라면 다음과 같이 입력됩니다.

user_name ='Jun'
demo_name ='model1'

In [ ]:

```
user_name = '' # 사용자의 이름
demo_name = '' # 저장한 모델의 이름
```

In [ ]:

```
knn = ColorClassifier(user_name=user_name)
train = knn.load_model(demo_name)
knn.fit("hsv")
```

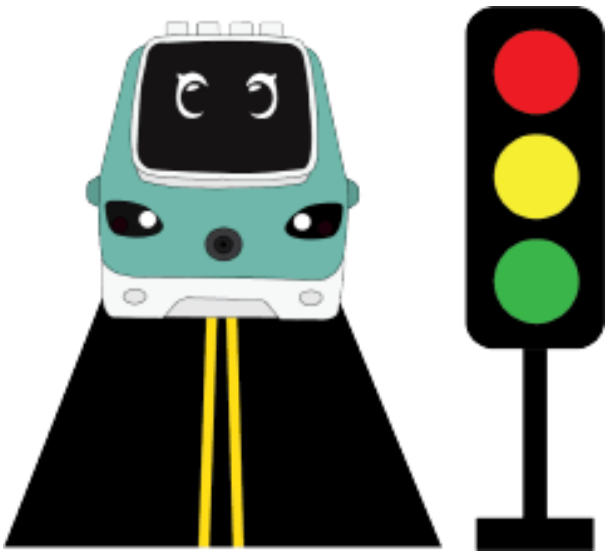
모델이 성공적으로 로드되면 테스트할 수 있습니다.
아래 코드를 실행하고 Zumi에게 보여줄 색상 카드를 선택하세요.
준비가 되면 Enter 키를 눌러 화면에서 결과를 확인합니다. "q"를 누르면 프로그램이 반복문에서 벗어나 카메라를 끕니다.

In [ ]:

```
camera.start_camera()

while True:
        user_input = input("예측하려면 'Enter'를 누르고 종료하려면 'q'를 입력하세요.")

        if user_input == "q":
            break

        image = camera.capture()
        predict = knn.predict(image)
        print(predict)

camera.close()
```

신호등

Zumi가 색상을 인식하도록 훈련시켰지만 Zumi는 아마도 보는 색상에 따라 결정을 내려야 할 것입니다!
신호등을 생각해보십시오. 신호등에는 세 가지 색상이 있습니다.
각 색상은 무엇을 의미합니까?



데이터 수집

색상 훈련 마법사로 돌아가 "빨간색", "노란색" 및 "녹색" 레이블을 훈련시킵니다.
그런 다음 모델을 "trafficlight"로 저장합니다.

결정하기

이 활동에서는 Zumi가 빨간색을 보면 멈추고, 초록색을 보면 운전하고, 노란색을 보면 느린 속도로 운전하게 합니다.
`stop()`, `forward(speed=30)` 및 `forward(speed =70)` 각각 빨간색, 노란색 및 녹색 일때의 행동입니다.

예를 들어:

```
if predict == "yellow":
    zumi.forward(speed=30)
```

모델 로드

따옴표 안에 유져 이름과 모델 이름(trafficlight)을 이름을 입력하여 프로젝트를 로드합니다.

In [ ]:

```
user_name = ''              # 사용자의 이름
demo_name = 'trafficlight'  # 저장한 모델의 이름

knn = ColorClassifier(user_name=user_name)
train = knn.load_model(demo_name)
knn.fit("hsv")
```

If 문 삽입

신호등 색상에 맞는 올바른 동작으로 코드를 작성하세요.
그런 다음 주미 컬러 카드를 보여주고 Enter 키를 눌러 주미가 반응하는지 테스트합니다!

In [ ]:

```
camera.start_camera()

try:
    while True:
        user_input = input("예측하려면 'Enter'를 누르고 종료하려면 'q'를 입력하세요.")

        if user_input == "q":
            break

        image = camera.capture()
        predict = knn.predict(image)
        screen.draw_text_center(predict)

        # 여기에 조건문을 추가하세요.

    camera.close()

finally:
    camera.close()
```

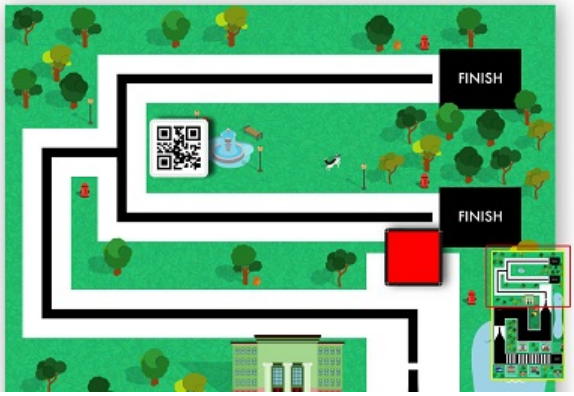
해결 방안

[해결 방안을보려면 클릭하세요!]

```
camera.start_camera()
try:
    while True:
        user_input = input("Press 'enter' to predict or 'q to quit: ")
        if user_input == "q":
            break
```

```
        image = camera.capture()
        predict = knn.predict(image)
        print(predict)
        if predict == "green":
            zumi.forward(speed=70)
        elif predict == "yellow":
            zumi.forward(speed=30)
        elif predict == "red":
            zumi.stop()
    camera.close()
finally:
    camera.close()
```

대회 : 어떻게 사용되나요?



맵의 세 번째 부분에서는 주미가 빨간색 카드일 때 멈춰야 하며, 인식했다는 의미로 '도레미'소리를 내야합니다.
또한, 녹색 카드로 교체될 때까지 정지 상태를 유지해야 합니다. 그리고 녹색 카드로 변경되면 주미는 다시 이동합니다.
신호등 활동에서 배운 것을 사용하여 주미가 빨간색에서 멈추고 녹색에서 진행하도록 프로그래밍 해 보세요!

다음은 빨간색을 인식하는 모델을 사용하여, 빨간색이 인식되기를 기다리는 예제입니다.
사용자명과 모델명은 예시이므로 자신의 사용자명과 직접 만든 모델명을 입력합시다.

In [ ]:

```
user_name = 'roy'        # 사용자의 이름
demo_name = 'challenge'  # 저장한 모델의 이름

knn = ColorClassifier(user_name=user_name)
train = knn.load_model(demo_name)
knn.fit()

# 카메라를 켜기
camera.start_camera()

# 빨강색을 찾을때까지 기다립니다.
while True:
    time.sleep(0.1)
    color = knn.predict((camera.capture()))
    print(color)

    if color == ('red'):
        print("빨간색이 인식되었다!")
        break

# 카메라 끄기
camera.close() # Turn off the camera.
```