import cv2

import numpy as np

# Load YOLO model and its configuration

yolo\_net = cv2.dnn.readNet('yolov3.weights', 'yolov3.cfg')

classes = []

with open('coco.names', 'r') as f:

classes = f.read().strip().split('\n')

# Set the target for weed class (modify class\_id according to your YOLO model)

weed\_class = "weed"

class\_id = classes.index(weed\_class)

# Step 1: Load and preprocess the dataset

def load\_dataset(dataset\_path):

# TODO: Load the dataset from the given dataset\_path and preprocess it

# You need to read the images and their corresponding labels from the dataset

# and return them as a list of tuples, e.g., [(image\_path, label), ...]

pass

# Step 2: Implement weed detection using YOLO

def detect\_weeds(image):

# Prepare the image for YOLO

blob = cv2.dnn.blobFromImage(image, 1/255.0, (416, 416), swapRB=True, crop=False)

yolo\_net.setInput(blob)

# Run forward pass through the network

detections = yolo\_net.forward()

weed\_mask = np.zeros(image.shape[:2], dtype=np.uint8)

for detection in detections:

for obj in detection:

scores = obj[5:]

class\_id = np.argmax(scores)

confidence = scores[class\_id]

if confidence > 0.5: # You can adjust the confidence threshold

if classes[class\_id] == weed\_class:

x, y, w, h = (obj[0:4] \* np.array([image.shape[1], image.shape[0], image.shape[1], image.shape[0]])).astype(int)

weed\_mask[y:y+h, x:x+w] = 255

return weed\_mask

# Step 3: Simulate the spraying process by printing the positions of the detected weeds

def spray\_weeds(weed\_mask):

for row in range(weed\_mask.shape[0]):

for col in range(weed\_mask.shape[1]):

if weed\_mask[row, col]:

print(f"Spraying weed at position ({col}, {row})")

# Step 4: Main function to perform crop and weed detection

def main():

dataset\_path = "path/to/your/dataset"

dataset = load\_dataset(dataset\_path)

for image\_path, label in dataset:

image = cv2.imread(image\_path)

weed\_mask = detect\_weeds(image)

spray\_weeds(weed\_mask)

# Run the main function

if \_\_name\_\_ == "\_\_main\_\_":

main()s