Egg Detection and Counting Using YOLOv11

🥚 Overview

This project demonstrates how to use **YOLOv11** to detect and **count eggs** in a video stream. It does not classify the eggs into cracked/uncracked or brown/white — only detects and counts the total number of eggs as they cross a defined line.

The model is trained on a custom dataset hosted on **Roboflow**, and it can be run both in Google Colab and locally on a Windows machine.

📁 Requirements

Install required libraries:

```
pip install ultralytics opencv-python roboflow
```

Make sure Python is added to your Windows PATH if running locally.

🎯 Dataset Preparation via Roboflow

1. Sign up or log in at https://universe.roboflow.com/
2. Generate a dataset version in **YOLOv11 format**
3. Copy your Roboflow **API key** from the dataset
4. Sample dataset: https://universe.roboflow.com/diplome/eggsdetect

Use the following code block to download the dataset:

```python
from roboflow import Roboflow
rf = Roboflow(api_key="YOUR_API_KEY")
project = rf.workspace("your-workspace").project("your-project")
version = project.version(1)
dataset = version.download("yolov11")
```

Training the Model

```python
from ultralytics import YOLO

model = YOLO("yolo11s.pt")  # Lightweight and fast
results = model.train(
    data=f"{dataset.location}/data.yaml",
    epochs=50,
    imgsz=640,
    batch=32,
    device="cuda",  # or "cpu" if GPU not available
    name="egg_detector",
    amp=True
)
```

How to Run the Egg Counting Program

- Place the trained weights at: `model/best.pt`
- Place the video file at: `video/your_video.mp4`

Use the following script (`main.py`) to detect and count eggs:

◇ Full Code: `main.py`

```python
import cv2
from ultralytics import YOLO
import os

# Allow duplicate OpenMP if needed
os.environ["KMP_DUPLICATE_LIB_OK"] = "TRUE"

# Load model
model = YOLO("model/best.pt")

# Video input
video_path = "video/your_video.mp4"
cap = cv2.VideoCapture(video_path)

# Read first frame for dimensions
ret, frame = cap.read()
if not ret:
    raise Exception("Failed to open video.")

frame = cv2.rotate(frame, cv2.ROTATE_90_CLOCKWISE)
height, width = frame.shape[:2]
fps = cap.get(cv2.CAP_PROP_FPS) or 30

# Output writer
out = cv2.VideoWriter("output_egg_counter.mp4", cv2.VideoWriter_fourcc(*"mp4v"),
fps, (width, height))
cap.set(cv2.CAP_PROP_POS_FRAMES, 0)

line_y = int(height * 0.6)
egg_count = 0
counted_centers = []
frame_number = 0

while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break
    frame = cv2.rotate(frame, cv2.ROTATE_90_CLOCKWISE)
    results = model.predict(source=frame, conf=0.5, verbose=False)
    boxes = results[0].boxes

    for box in boxes:
        x1, y1, x2, y2 = map(int, box.xyxy[0].tolist())
        cx = int((x1 + x2) / 2)
        cy = int((y1 + y2) / 2)

        if line_y - 5 <= cy <= line_y + 5:
            if cx not in counted_centers:
                egg_count += 1
                counted_centers.append(cx)

        cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)
        cv2.circle(frame, (cx, cy), 5, (255, 0, 0), -1)

    cv2.line(frame, (0, line_y), (width, line_y), (0, 0, 255), 2)
    cv2.putText(frame, f"Egg Count: {egg_count}", (20, 40),
                cv2.FONT_HERSHEY_SIMPLEX, 1.2, (0, 255, 255), 2)

    out.write(frame)
    frame_number += 1

cap.release()
out.release()
print(f"Total eggs counted: {egg_count}")
```

Output

- You will get a video file named: `output_egg_counter.mp4`
- It will have green boxes on each egg and a running count on screen

You're now ready to detect and count eggs automatically from video!

Need to deploy this to Raspberry Pi or optimize it for live camera? Just ask!