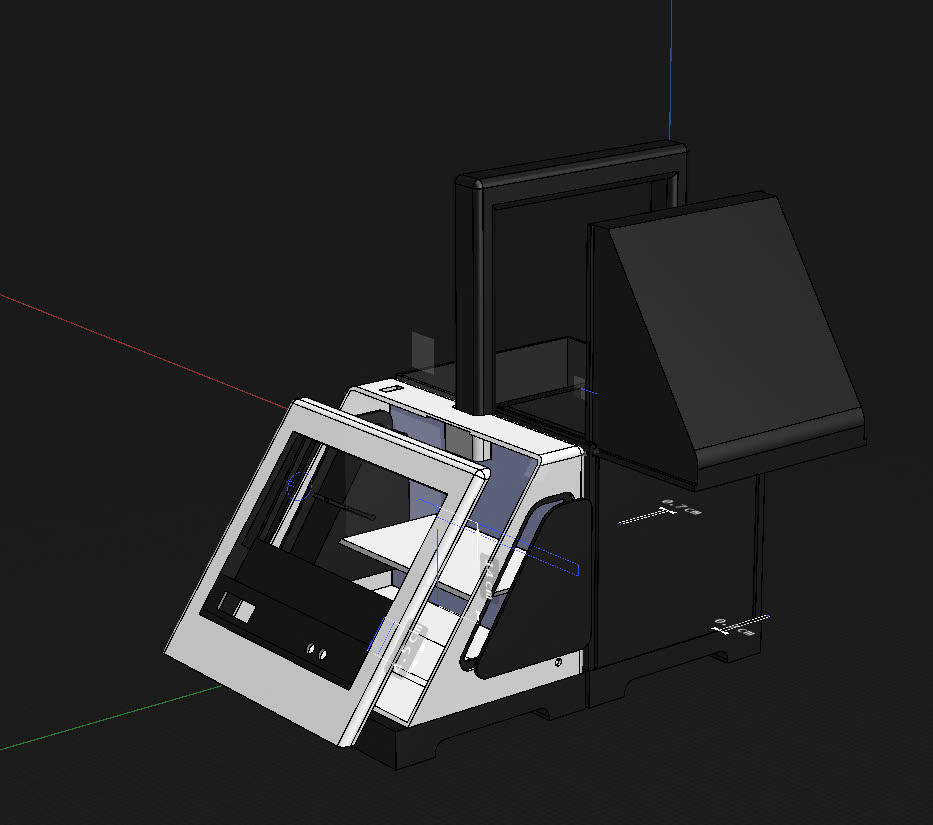
Buồng tối và sáng: diện tích các buồng là bao nhiêu, sử dụng camera gì, thuật toán để tính đo lường các chỉ số như thế nào...



# Thiết kế vỏ mô hình

Buồng sáng:

* Mica trong suốt – dày 5cm
* Kích thước: 30x20x30cm DxRxC

Buồng tối:

* Nhựa đen rỗng – dày 5cm
* Kích thước: 30x20x30cm DxRxC

# Camera :PiCamera

PiCamera là một module camera được thiết kế đặc biệt để sử dụng với Raspberry Pi. Nó kết nối trực tiếp với Raspberry Pi qua cổng CSI (Camera Serial Interface), cho phép chụp ảnh và quay video chất lượng cao.

PiCamera là một module camera mạnh mẽ và linh hoạt, nâng cao khả năng của Raspberry Pi. Cho dù cho các dự án sở thích, mục đích giáo dục hay ứng dụng chuyên nghiệp, PiCamera cung cấp khả năng chụp ảnh và quay video chất lượng cao, dễ dàng tích hợp và có rất nhiều tiềm năng sử dụng, khiến nó trở thành một bổ sung có giá trị cho bất kỳ thiết lập Raspberry Pi nào.

# Thuật toán và khởi tạo model

Cài đặt môi trường: Đầu tiên, cài đặt các thư viện cần thiết như PyTorch, OpenCV và các công cụ hỗ trợ khác. Điều này có thể được thực hiện bằng cách tạo một môi trường ảo và cài đặt các gói cần thiết.

Nạp dữ liệu: Sử dụng một trình nạp dữ liệu để nạp dữ liệu huấn luyện và kiểm tra. Trình nạp dữ liệu giúp quản lý và nạp dữ liệu theo từng batch, tối ưu hóa quá trình huấn luyện.

Xây dựng mô hình YOLOv11: Khởi tạo mô hình YOLOv11 với kiến trúc phù hợp hoặc tải mô hình YOLOv11 đã được huấn luyện trước để tinh chỉnh trên dữ liệu của bạn.

Định nghĩa hàm mất mát và tối ưu hóa: Chọn hàm mất mát và thuật toán tối ưu hóa phù hợp. YOLO thường sử dụng hàm mất mát kết hợp giữa mất mát định vị, mất mát độ tin cậy và mất mát xác suất lớp.

Huấn luyện mô hình: Thực hiện quá trình huấn luyện qua nhiều epoch. Trong mỗi epoch, mô hình sẽ dự đoán các hộp giới hạn, so sánh chúng với nhãn thực để tính toán mất mát và cập nhật trọng số tương ứng.

Đánh giá mô hình: Sau khi huấn luyện, sử dụng tập kiểm tra để đánh giá độ chính xác của mô hình. Các chỉ số đánh giá bao gồm Precision, Recall và Mean Average Precision (mAP).[8]

A screenshot of a computer

Description automatically generated

Hình 2‑9 Quá trình training

1. Chương trình Heatmap

Heatmap (bản đồ nhiệt) là một phương pháp trực quan hóa dữ liệu hai chiều, hiển thị cường độ của các giá trị riêng lẻ trong một tập dữ liệu dưới dạng màu sắc. Các màu sắc này có thể thay đổi theo sắc thái hoặc cường độ. Trong một số ứng dụng, như phân tích tội phạm hoặc theo dõi nhấp chuột trên trang web, màu sắc biểu thị mật độ các điểm dữ liệu hơn là các giá trị cụ thể liên quan đến mỗi điểm.

*Tạo Heatmap*

Tạo dự đoán

Sau khi mô hình YOLOv11 đã được huấn luyện, bước tiếp theo là sử dụng mô hình để tạo ra các dự đoán trên một tập dữ liệu kiểm tra hoặc dữ liệu mới. Các dự đoán này bao gồm các hộp giới hạn và xác suất của các đối tượng được phát hiện trong hình ảnh.

Xây dựng Heatmap

Dựa trên các dự đoán của mô hình, chúng ta có thể xây dựng một heatmap. Heatmap là một biểu đồ thể hiện mật độ hoặc tần suất xuất hiện của các đối tượng trong một hình ảnh hoặc tập hình ảnh. Để tạo heatmap, chúng ta chia hình ảnh thành các ô lưới và đếm số lượng hộp giới hạn rơi vào mỗi ô lưới. Sau đó, chúng ta biểu diễn các ô lưới này bằng các màu sắc tương ứng với tần suất xuất hiện của đối tượng.

Trực quan hóa Heatmap

Sử dụng các công cụ trực quan hóa dữ liệu như Matplotlib hoặc Seaborn để vẽ heatmap. Heatmap giúp dễ dàng nhận diện các vùng trong hình ảnh mà mô hình phát hiện đối tượng thường xuyên, giúp đánh giá hiệu suất của mô hình và xác định các vùng quan trọng trong hình ảnh.

*Triển khai mô hình*

Chọn nền tảng triển khai

Để triển khai mô hình YOLOv11 trong các ứng dụng thực tế, cần chọn một nền tảng phù hợp như server, dịch vụ đám mây hoặc thiết bị nhúng. Các dịch vụ đám mây như AWS, Google Cloud hoặc Microsoft Azure có thể cung cấp tài nguyên tính toán mạnh mẽ để triển khai mô hình.

Tích hợp mô hình

Sau khi chọn nền tảng, tích hợp mô hình vào ứng dụng hoặc hệ thống. Điều này có thể bao gồm xây dựng API để nhận hình ảnh và trả về kết quả dự đoán hoặc tích hợp mô hình vào một ứng dụng web hoặc di động.

Tối ưu hóa hiệu suất

Để mô hình hoạt động hiệu quả trong các ứng dụng thực tế, cần tối ưu hóa nó để giảm thời gian xử lý và yêu cầu tài nguyên. Các kỹ thuật như pruning, quantization và sử dụng phần cứng tăng tốc như GPU hoặc TPU có thể giúp cải thiện hiệu suất..

*Những kiến thức nền tảng*

Xử lý ảnh

Kiến thức về xử lý ảnh bao gồm các kỹ thuật tiền xử lý như thay đổi kích thước hình ảnh, chuẩn hóa dữ liệu và áp dụng các phương pháp tăng cường dữ liệu. Những kỹ năng này rất quan trọng để chuẩn bị dữ liệu đầu vào cho mô hình.

Học sâuHiểu biết về các khái niệm cơ bản trong học sâu như các loại mạng neural, hàm mất mát và các thuật toán tối ưu hóa. Nắm vững các khái niệm này giúp thiết kế và huấn luyện mô hình hiệu quả.

Kiến trúc YOLO

Hiểu về kiến trúc của các phiên bản YOLO, từ YOLOv1 đến YOLOv11, bao gồm các cải tiến và khác biệt giữa các phiên bản. Kiến thức này giúp chọn phiên bản phù hợp và tùy chỉnh mô hình cho các tác vụ cụ thể.

Công cụ và framework

Thành thạo trong việc sử dụng các công cụ và framework như PyTorch, TensorFlow, OpenCV và các thư viện trực quan hóa dữ liệu như Matplotlib và Seaborn. Những công cụ này hỗ trợ phát triển, huấn luyện và đánh giá mô hình hiệu quả.

A mouse in a cage

Description automatically generated