Report

- 方法介紹

  在這次的期末專案中，我們使用 OpenCV 載入 YoloV3 模型，實現 Object Detection。

  首先，在載入模型的部分，因為模型為 Darknet 格式，因此可以透過 OpenCV 中的 readNetFromDarknet 函式來建立模型並讀取預訓練參數。在 2-1 中，為了能夠辨識出更多的物件，我們使用正常大小的 Yolov3 模型，而在 2-2 中，因為考量到 Performance 的問題，我們使用 Yolov3-Tiny 模型。這些模型的參數都可以在論文的網站中找到。

  接著，我們會讀取圖片 (imread) 並對圖片進行前處理 (blobFromImage)，將圖片進行標準化，並轉為符合模型的輸入格式。透過 net.forward() 將圖片輸入到模型中進行 inference。

  經過 forward pass 後，模型會輸出非常多的 bounding box，然而不是每一個 bounding box 中都會有物件，因此需要對這些 bounding box 進行後處理。在 postprocess 函式中，我們會依序檢查每一個 bounding box 的以下資訊:「包含物件的信心程度」、「每一個類別的機率」、「X 座標」、「Y 座標」、「寬度」以及「長度」。

  只有當「包含物件的信心程度」大於 0.5 時，我們才會保留這個 bounding box。經過這個步驟後，一個物件將會被多個 bounding box 給匡起來，因此我們還會進行 non-maximum suppression (threshold = 0.4)，將多餘的 bounding box 給去除。最後，再將留下來的 bounding box 根據其座標以及尺寸繪製在圖片上，並且附上預測的物體類別。

  當模型的輸出經過上述的後處理步驟後，我們會透過之前的 Lab 的方法，將最終的圖片寫入到 framebuffer 中，將結果顯示於螢幕上。

- 挫折與挑戰

在這次的專案中，我們原先是使用 OpenCV 讀取 Yolov5 的 ONNX 檔案，但是因為版本問題，嘗試了非常多的辦法始終還是無法成功。最後，我們針對 Yolo 系列的模型經過更多的嘗試與研究，發現在 Yolov3 所使用的 Darknet 格式下，能夠與嵌入式裝置上的 OpenCV 3.4 成功相容。