高等電腦視覺

Final Project

姓名:闕楷宸
學號:114318047
指導老師:黃正民

作業說明:

作業架構:

```
best_mix_150.pt                    requirement.txt
boat_direction_comparison.png      SD_Dreambooth
boat_direction_yolo_n_tradition.py SDXL_Dreambooth
Dataset_given                      stern_wave_comparison_table.png
fp_per_image_comparison.png        stern_wave_yolo_n_tradition.py
output                             tools
problem.pdf                        Unit_test
README.md                          YOLOV8_OBB
```

requirement.txt:
請在執行程式前先設置環境
pip install -r requirement.txt

best_mix_150.pt: YOLOv8 OBB model for AI method in stern wave and boat direction detection

boat_direction_yolo_n_tradition.py:
執行船的方向程式於傳統影像處理與AI

stern_wave_yolo_n_tradition.py:
執行尾流偵測程式於傳統影像處理與AI

輸出表格:
stern_wave_comparison_table.png
boat_direction_comparison.png
fp_per_image_comparison.png

輸出圖片及影像:
output/

```
boat_direction_trad_vis  stern_wave_trad_vis
boat_direction_yolo_vis  stern_wave_yolo_vis
```

主程式boat_direction_yolo_n_tradition.py與stern_wave_yolo_n_tradition.py

建置執行(window,linux)

建議在虛擬環境中執行:

```
python3 -m venv acv_env
source acv_env/bin/activate
pip install --upgrade pip
pip install -r requirement.txt
```

Source code:

https://github.com/KaiChenChueh/ACV_Final.git

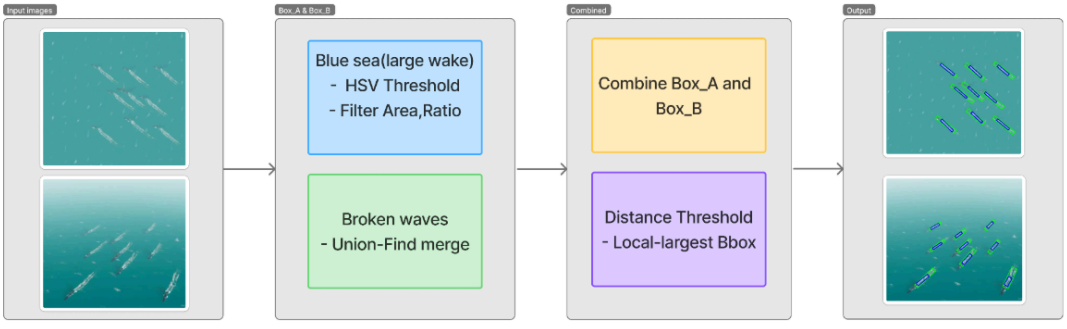
1. Stern wave detection

Use the same program and parameters for all photos

Comparison: Traditional (T) vs YOLO (Y)
Metric: Avg IoU / Exec Time

	1	2	3
a	T: 0.41 / 0.009s Y: 0.37 / 0.910s	T: 0.38 / 0.005s Y: 0.33 / 0.007s	T: 0.72 / 0.098s Y: 0.36 / 0.040s
b	T: 0.38 / 0.008s Y: 0.31 / 0.008s	T: 0.43 / 0.017s Y: 0.31 / 0.007s	T: 0.31 / 0.015s Y: 0.53 / 0.006s
c	T: 0.13 / 0.149s Y: 0.20 / 0.008s	T: 0.48 / 0.424s Y: 0.39 / 0.006s	T: 0.13 / 0.626s Y: 0.54 / 0.006s

stern_wave_comparison_table.png



Discussion(1)

本程式由傳統影像處理以及YOLOv8 OBB所組成，上半部為傳統影像處理的部份，以下說明:

傳統影像處理:

Candidate Bbox Generation:

- **Box_A (顏色特徵)**：使用 **`get_blue_sea_boxes`** 透過 HSV 閾值過濾出深藍色海浪區域(low saturation, high value)。
- **Box_B (紋理特徵)**：使用 **`get_loop_and_micro_boxes`** 透過自適應閾值 (Adaptive Thresholding) 捕捉細碎波浪與紋理。

Refinement:

- **Union-Find** 合併演算法：使用 **`merge_overlapping_boxes_unionfind`**，利用演算法將重疊或鄰近的碎片化框合併為完整物件。
- 雜訊抑制：**`suppress_close_small_boxes`** 函數會移除靠近主要波浪的小型雜訊框。

深度學習流程:

模型載入：載入預訓練權重 **`best_mix_150.pt`**

推論與格式轉換：執行 YOLOv8 推論，並將輸出的 OBB (Oriented Bounding Box) 格式轉換為與傳統方法一致的座標格式以便評估。

以下則為視覺化的部份(表格輸出):

解析 **Ground Truth (`parse_gt_file`)**：讀取標註文件中的座標並轉換為多邊形點集。

計算 **IoU (`calculate_iou_poly`)**：使用 **`shapely`** 函式庫計算兩個旋轉多邊形之間的 Intersection over Union (IoU)。

評估指標 **(`compute_avg_iou`, `compute_tp_fp_fn`)**：計算平均 IoU，並根據 IoU 閾值（預設 0.3）統計 True Positive (TP)、False Positive (FP) 與 False Negative (FN)。

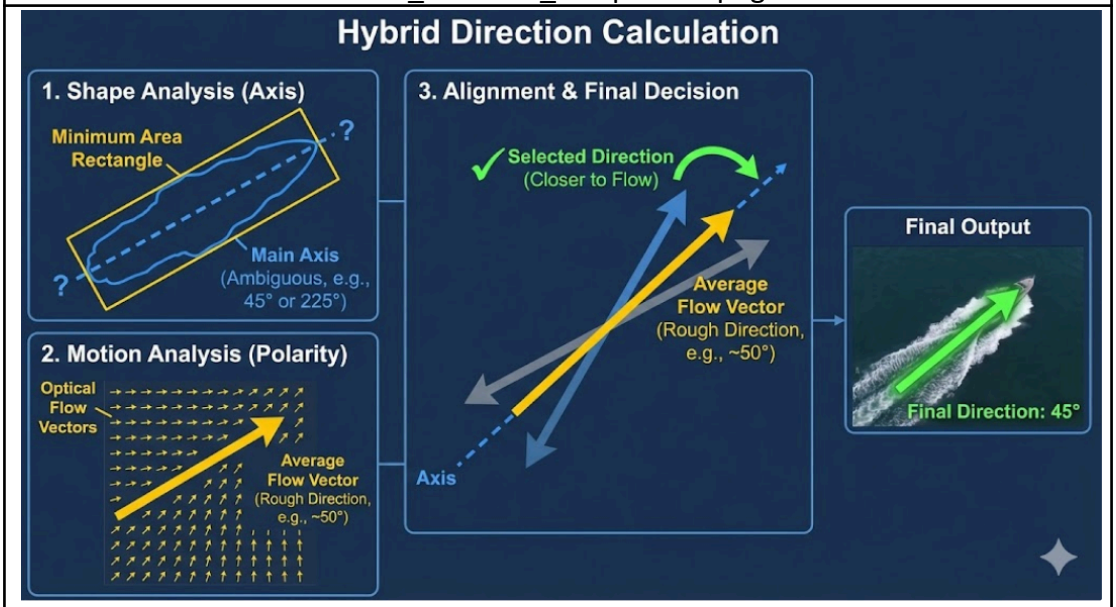
2.Identify the boat direction i image sequence

Use the same program and parameters for all videos

Comparison: Avg Angle Error | Tracking Rate | Avg Time/Frame

	1			2		
d	T: 5.9°	13.5%	0.007s	T: 40.4°	47.1%	0.025s
	Y: 5.5°	100.0%	0.011s	Y: 2.9°	42.9%	0.013s
e	T: 4.1°	13.1%	0.005s	T: 95.9°	90.0%	0.026s
	Y: 6.5°	100.0%	0.008s	Y: 41.0°	30.0%	0.010s
f	T: 56.9°	11.4%	0.024s	T: 45.2°	52.2%	0.026s
	Y: 13.9°	65.0%	0.010s	Y: 40.9°	72.1%	0.010s

boat_direction_comparison.png



Discussion (2)

此程式碼可分成四個模組：

資料處理與工具模組 **(Utility & Data Handling)**

- **GT** 解析 **(parse_gt_file)**：讀取Ground Truth文字檔，解析每一幀船隻的頭尾座標 **(p1, p2)**。
- 幾何計算：包含計算角度 (**calculate_gt_angle**) 與計算角度誤差 (**get_angle_diff**，處理 0°/360° 的週期性問題)。

方法一：傳統電腦視覺演算法 **(Traditional Approach)**

這是由多個步驟串聯而成的 Pipeline：

- 偵測階段 **(TradBoatDetector)**：採用多種特徵融合策略。
  - 顏色特徵：利用HSV色彩空間過濾（高飽和度、低亮度）來抓取船體。
  - 特定場景特徵 **(Sunset)**：針對夕陽/反光場景，利用藍色通道閾值與Top-Hat運算。
  - 動態特徵：若靜態特徵失效，使用背景相減法（**BackgroundSubtractorMOG2**）抓取移動物體。
- 光流計算 **(FlowComputer)**：
  - 使用 Lucas-Kanade 光流法（**calcOpticalFlowPyrLK**）計算船隻區域的運動向量，用於輔助判斷前進方向。
- 追蹤與方向估計 **(HybridTracker)**：
  - 追蹤：基於歐式距離將當前偵測結果與歷史軌跡關聯。
  - 方向計算：
    1. 主軸提取：對船隻輪廓進行PCA或最小外接矩形（minAreaRect）找出船身主軸。
    2. 方向修正：利用光流向量解決PCA的180度模糊問題（區分船頭與船尾）。

方法二：深度學習演算法 **(YOLO Approach)**

- 模型推論 **(yolo_detect_centers)**：
  - 載入預訓練的 YOLOv8 模型（**best_mix_150.pt**）。
  - 直接輸出 **OBB (Oriented Bounding Box, 旋轉邊框)**。
- 方向計算：
  - 直接從 YOLO 預測的旋轉框座標（**xyxyxyxy**）計算長邊的角度作為航向。

評估與視覺化模組 **(Evaluation & Visualization)**

- 雙線執行：針對每個測試影片（**d1, d2, e1...**）分別執行上述兩種方法。
- 指標計算：
  - 角度誤差 **(Avg Angle Error)**：預測角度與 GT 角度的差異。
  - 追蹤率 **(Tracking Rate)**：成功偵測到船隻的幀數比例。
  - 執行速度 **(Time/Frame)**：計算平均每幀處理時間。
- 結果輸出：
  - 生成帶有箭頭視覺化的影片。
  - 使用 **matplotlib** 繪製詳細的比較表格（**boat_direction_comparison.png**），並以紅字標示表現較優的指標。