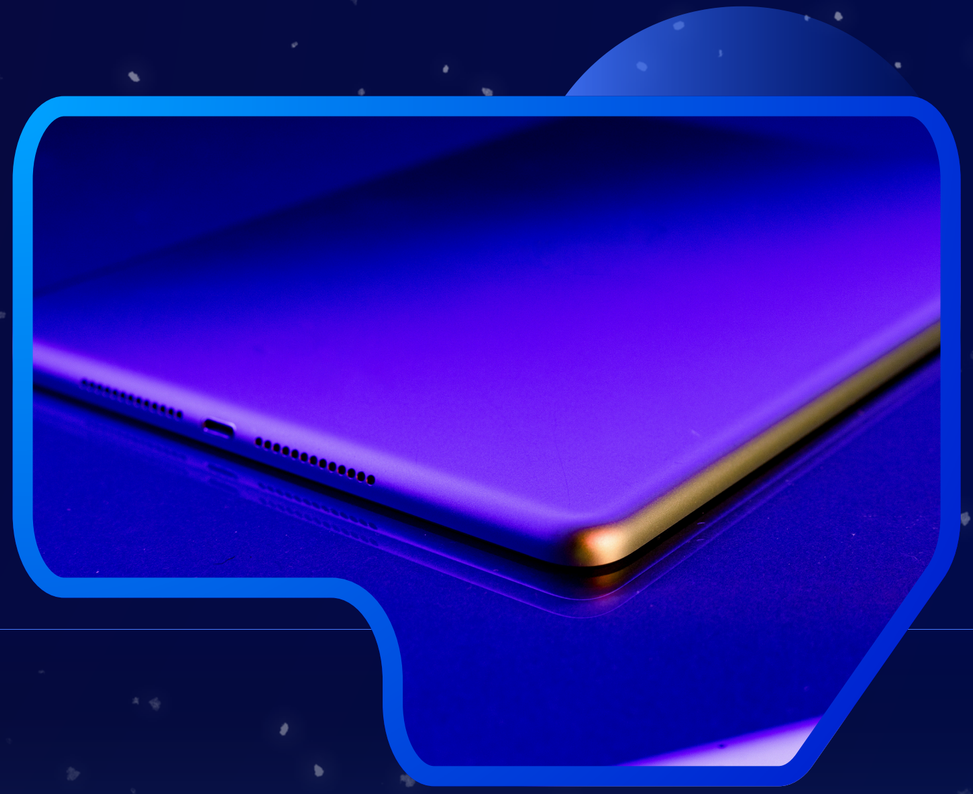
VISION-BASED VEHICLE ENTRY-EXIT TRACKING

Latar Belakang



Traditional vehicle entry systems face significant bottlenecks: manual verification is labor-intensive and prone to error, while RFID solutions struggle with adoption barriers due to user costs. This project leverages rapid advancements in Deep Learning. To bridge this gap, offering a non-intrusive, automated alternative that utilizes existing camera infrastructure to deliver high-speed vehicle identification without requiring any additional hardware from the driver.

Dataset

Indonesian License Plate Detection

Indonesian License Plate Detection

Source : Roboflow Universe

IMG_20230408_133259_flipped.jpg | IMG_20230408_133132_flipped.jpg | IMG_20230408_133220_flipped.jpg

Preprocessing

- Images resized to 640x640 pixels (Stretch) to match model input.

Augmentation Strategy

- Rotation: ±15°
- Shear: ±10° (Horizontal & Vertical)
- Brightness: ±20%
- Blur: Up to 0.8px
- Noise: Up to 1% of pixels

Model

YOLOv11n

- Method: Transfer Learning (using pre-trained weights).
- Benefit: Reduces computational cost & training time while maintaining high accuracy.

- Two-Stage Detection Strategy
- Model A (Vehicle Detector): Focuses solely on classifying and localizing vehicles.
- Model B (Plate Detector): Specialized instance optimized for detecting small, rectangular license plates.

fast-plate-ocr

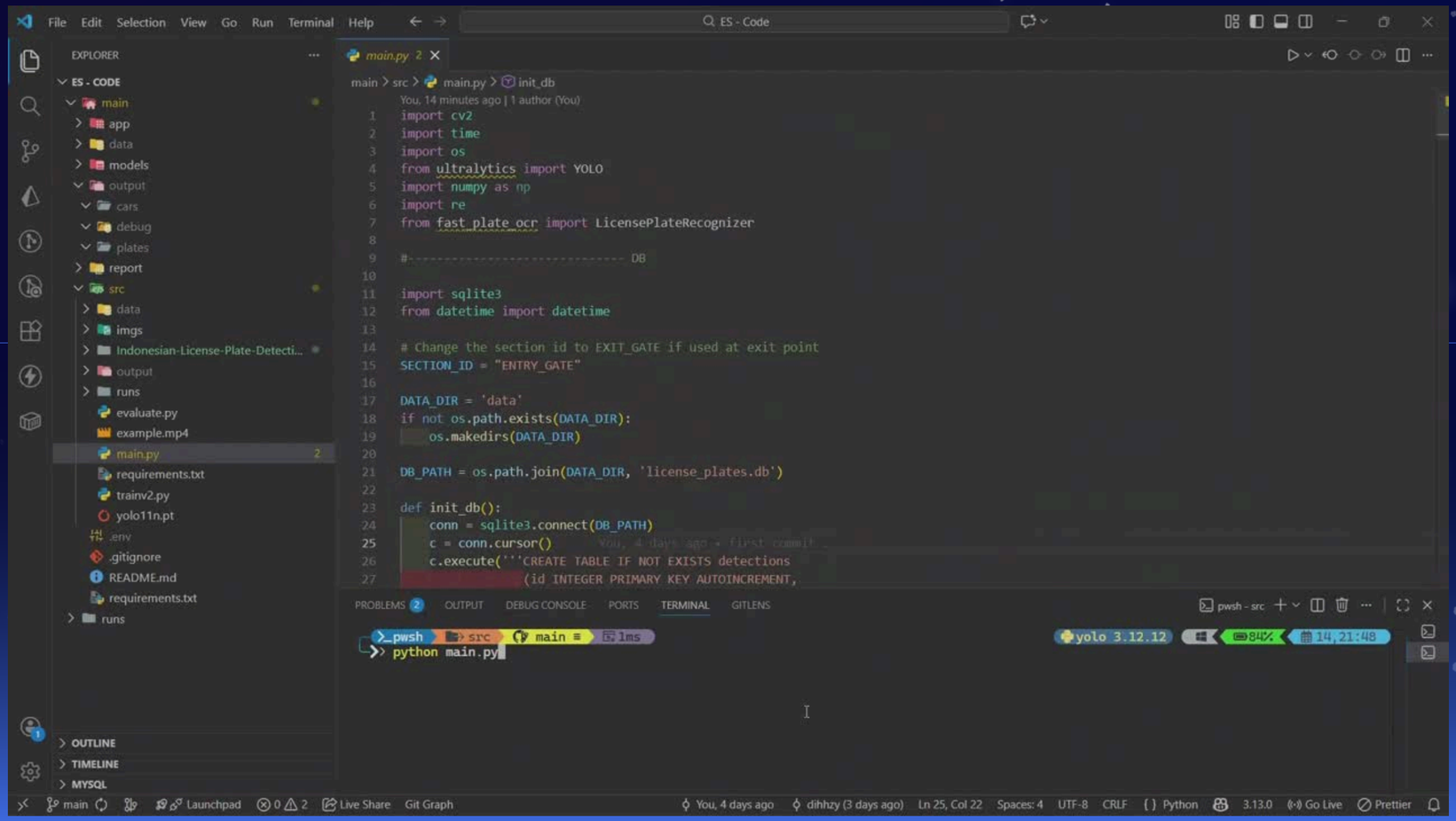
- Tool: FastPlateOCR (ONNX-based).
- Process: Receives the cropped plate region from Model B and extracts the alphanumeric text.

Evaluation

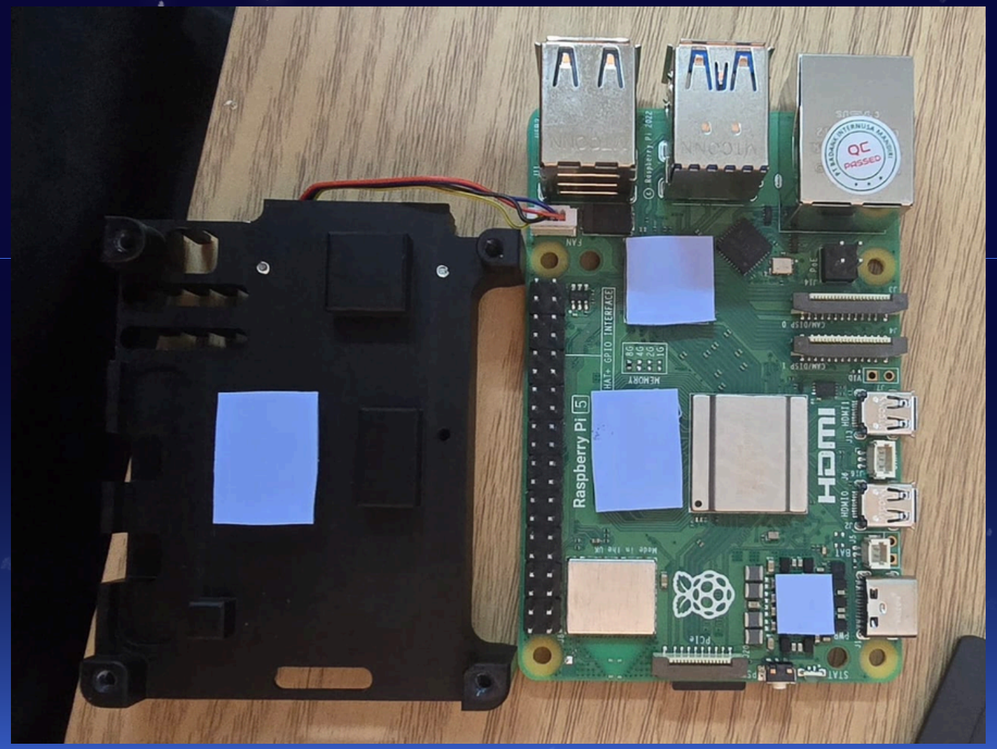
```
Evaluation Results:
mAP50: 0.9946
Precision: 0.9764
Recall: 1.0000
```

A manual evaluation was conducted for the ocr. On a test set of 44 images under suboptimal conditions. The system achieved perfect text recognition in 12 instances, while the remaining samples exhibited either minor discrepancies of 1-2 characters or complete detection failures. These results suggest that controlling environmental variables, particularly camera angle, is critical for maximizing OCR accuracy.

Demo


File Edit Selection View Go Run Terminal Help
ES - Code
EXPLORER
ES - CODE
main
app
data
models
output
cars
debug
plates
report
src
data
imgs
Indonesian-License-Plate-Detecti...
output
runs
evaluate.py
example.mp4
main.py
requirements.txt
trainv2.py
yolo11n.pt
.env
.gitignore
README.md
requirements.txt
runs
main.py 2
main > src > main.py > init_db
You, 14 minutes ago | 1 author (You)
import cv2
import time
import os
from ultralytics import YOLO
import numpy as np
import re
from fast_plate_ocr import LicensePlateRecognizer
#---------------------------- DB
import sqlite3
from datetime import datetime
# Change the section id to EXIT_GATE if used at exit point
SECTION_ID = "ENTRY_GATE"
DATA_DIR = 'data'
if not os.path.exists(DATA_DIR):
os.makedirs(DATA_DIR)
DB_PATH = os.path.join(DATA_DIR, 'license_plates.db')
def init_db():
conn = sqlite3.connect(DB_PATH)
c = conn.cursor()
c.execute('''CREATE TABLE IF NOT EXISTS detections
(id INTEGER PRIMARY KEY AUTOINCREMENT,
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
PORTS
TERMINAL
GITLENS
pwsh - src
_pwsh src main ≡ lms
yolo 3.12.12
84%
14, 21:48
python main.py
OUTLINE
TIMELINE
MYSQL
main
Launchpad
0 2
Live Share
Git Graph
You, 4 days ago
dihhzy (3 days ago)
Ln 25, Col 22
Spaces: 4
UTF-8
CRLF
Python
3.13.0
Go Live
Prettier

Demo


Raspberry Pi 5
HDMI
QC PASSED

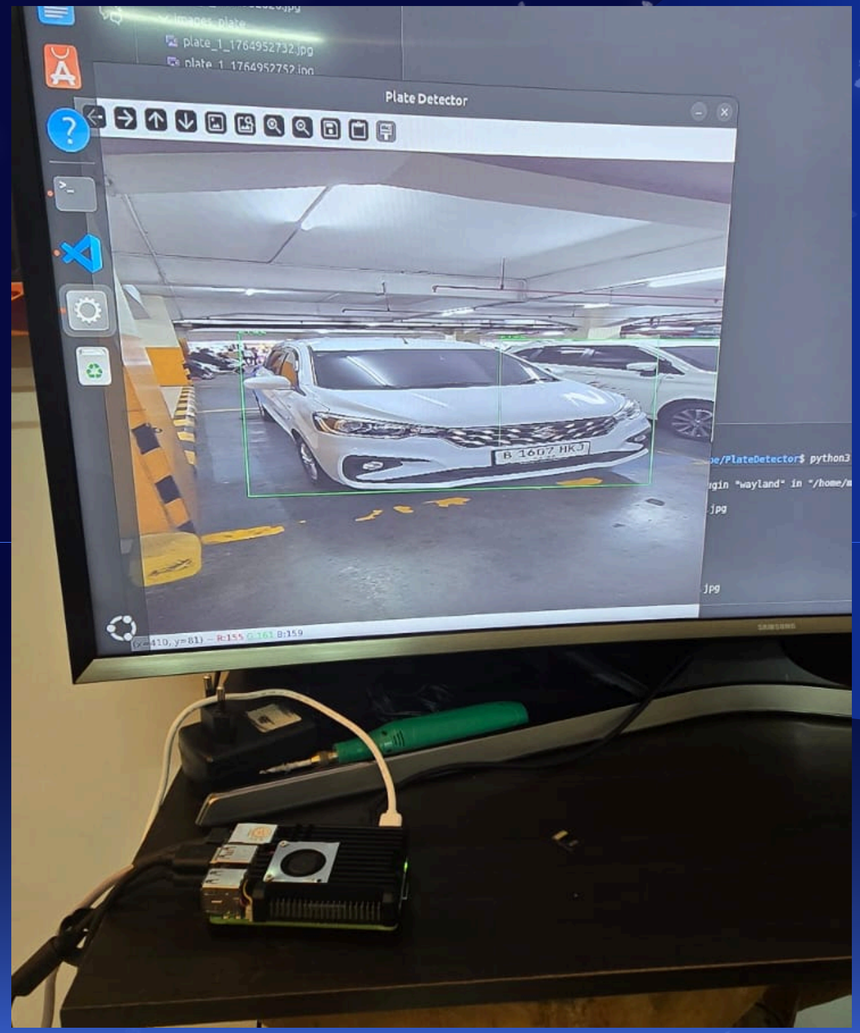


Plate Detector
plate_1_1764952752.jpg
B 1607 HKJ

Reflection

- High detection metrics (mAP = 0.99) do not guarantee end-to-end success.
- The OCR stage proved highly sensitive to input quality; low resolution and extreme angles caused the majority of failures in the test set (12/44 success rate).
- Software optimization cannot fully compensate for poor physical deployment.

Future Technical Roadmap

- Perspective Transformation: Implement mathematical correction to "flatten" angled plates before OCR processing.
- Dataset: Define physical installation standards to align real-world inputs with training data conditions.

Thank You