UNIVERSITI MALAYSIA PAHANG
AL-SULTAN ABDULLAH

BSD2513 ARTIFICIAL INTELLIGENCE
GROUP PROJECT

TITLE :
UMPSA VEHICLE REGISTRATION STICKER DETECTION

LECTURER :
DR KU MUHAMMAD NA'IM BIN KU KHALIF

PREPARED BY :
GROUP I

NAME	MATRIC ID	SECTION
NURUL FAQIHAH BINTI MAZLI AMRAN	SD22047	01G
NUR NABILA BINTI ABD RAHMAN	SD22037	
NUR NABILAH BINTI SUZELAN AMIR	SD22053	
NUR A'RIFAH AKMAL BINTI HUSSIN	SD22032	
MUHAMMAD DANISH AIMAN HARISS BIN ROSLI	SD22008	

TABLE OF CONTENT

1.0 EXECUTIVE SUMMARY

1.1 Description of the Selected Project

Vehicle registration has been critical in managing traffic and ensuring security within the University of Malaysia Pahang Student Association (UMPSA) campus. They have become a requirement and have been suggested as a key strategy by university authorities to lower the rate of unregistered vehicles. Thus, individuals are recommended to register their vehicles during their tenure at the university, particularly those who use the university's parking facilities, not only to maintain order but also to ensure the safety and security of the campus to some extent. As a result, vehicle registration is important for protecting the university's resources, reducing the risk of unauthorized vehicle access, and providing additional control in certain circumstances. Having a registered vehicle, for example, is an effective preventive practice that helps protect security and orderliness in the university setting. However, in crowded parking lots, it is difficult to manually identify vehicles that are not registered, making prompt and proactive steps to prevent unauthorized access and preserve campus security challenging.

To address this issue, real-time video analysis using image recognition and artificial intelligence may detect vehicles that do not display registration stickers or incorrectly, ensuring that vehicles on campus are registered. Vehicle Registration Sticker Detection is an artificial intelligence (AI) analytics solution that uses algorithms and deep learning techniques to distinguish between vehicles that display registration stickers and those that do not. The technology evaluates elements of a vehicle's registration sticker to quickly identify vehicles that aren't registered, even in crowded parking lots, while machine learning and/or machine learning-built reference models work behind the scenes to ensure proper processing. This information may subsequently be delivered in real-time to security professionals for immediate action, or it can be included in periodic reports to assess program and policy compliance. Vehicle Registration Sticker Detection has the benefit of automating and simplifying the procedure; for example, it eliminates the need for manual identification of whether vehicles are registered in large parking lots.

Previous research has not concentrated on the difficulty of establishing a single model for identifying both picture and real-time video images with and without registration stickers. All experiments were conducted using either live video or picture imagery. Manual vehicle registration detection is frequently challenging in achieving system scalability and automation. This is because approaches may not satisfy the requirements when vehicle traffic increases or the demand for constant monitoring and detection grows, necessitating more efficient and intelligent solutions. Therefore, the primary objective of our research is to create a method that can not only distinguish whether a vehicle is registered in photos and videos but also detect registration

stickers in videos in real-time, with fewer streaming parameters, lower computational cost, faster computational speed, and higher model accuracy to solve these problems.

1.2 Problem to be Solved

The University of Malaysia Pahang Al-Sultan Abdullah (UMPSA) faces a few challenges related to vehicle management on the campus. We want to solve the problem of detecting vehicle stickers for students and staff, enhance security, and avoid heavy traffic at the gate.

Firstly, the detection of vehicle stickers for students and staff. The system that we develop must accurately detect the UMPSA registration stickers. These stickers indicate whether the vehicles belong to a student residing on campus, a student living off campus, or a UMPSA staff member. This detection system will verify whether the vehicles are registered and distinguish between valid registration stickers and expired or unregistered ones. The challenge to solve this problem is to develop a machine-learning model capable of identifying the presence or absence of these stickers under various conditions, such as different lighting, vehicle models and sticker placement.

In addition, we want to enhance UMPSA security by ensuring all vehicles are registered and maintaining a safe environment on the campus. The automated system that detects unregistered vehicles in real-time will significantly increase campus security by preventing unauthorized access and ensuring that only valid registration vehicles are allowed to enter the campus. This system is reliable since it can function effectively in various scenarios.

Lastly, we want to solve the heavy traffic at the gate. The current manual process of checking vehicle registration can cause significant delays and traffic congestion at the campus gates, particularly during peak hours when cars in and out. An automated system that quickly processes vehicle entries and exits will help to reduce traffic and ensure smooth flow of vehicles. Minimizing the vehicle's delay and preventing the heavy traffic at the gate is crucial and needs to be solved.

Thus, our project aims to solve the problem that UMPSA is facing by developing an automated Vehicles Registration Sticker Detection System to address these issues. The system will leverage the Artificial Intelligence and image recognition technologies to identify the registered vehicles based on their sticker. We aim to have a well trained model with a comprehensive dataset of vehicles images with and without registration stickers to achieve high accuracy in detecting the presence of the registration stickers. We aim to have an efficient algorithm and optimization to enable real-time detection of registration stickers as vehicles enter or exit the campus. By implementing this system, we will enhance the campus security and reduce the traffic.

1.3 Basic Description of the Selected Dataset

The dataset for this project includes images of vehicles classified according to three specific registration stickers: stickers for students staying in UMPSA hostels, stickers for students stay outside UMPSA hostels and stickers for staff. Vehicles without these stickers or with different stickers are considered unregistered. In addition, the dataset contains another unregistered sticker to facilitate controlled testing. The main purpose of the database is to allow the model to verify vehicle registration by identifying one of these three stickers. The images show clearly visible decals attached to the windshield or other specific vehicle parts, taken in different conditions such as different lighting, angles and decal positions to ensure the generality of the model. It includes a variety of vehicles such as cars and motorcycles to cover all possible scenarios on a college campus.

The images are in a standard format, such as JPEG or PNG, with a high enough resolution to clearly identify the label, and at the same time optimized to reduce the computer load.Each image is labeled to show if a valid registration sticker is present, and for stickers, the location of the sticker is marked for easy object identification. The dataset is balanced according to the number of images in each class to avoid bias during model training. Image sources include university entry points where vehicle registrations are typically verified, online databases and publicly available vehicle image databases to ensure appropriate annotation and relevance to the project. In addition, project members can manually collect images to cover specific cases or conditions not found in existing data. Most images are used for model training, some validate the model during training to avoid parameter tuning and overfitting, and a separate set tests the performance of the final model to ensure that it generalizes well to unseen data. To complete the dataset, there is also an image in the background for the user interface. This comprehensive dataset ensures that the model focuses on verifying vehicle registration based on only three specific stickers and provides accurate results indicating whether or not a vehicle is registered based on those criteria.

Our dataset may be found at the following Google Drive folder : Dataset project

2.0 SUMMARY OF THE PROJECT CONTEXT AND OBJECTIVES

2.1 Summary of the Project Context

The project addresses the challenge of unregistered vehicles at the University of Malaysia Pahang Student Association (UMPSA) campus by developing an automated vehicle registration sticker detection system. This system uses image recognition and artificial intelligence to identify registered vehicles based on their license plates. Installed at all entrance and exit points of the university, including the main gate, the system automatically detects the presence of a registration sticker on each passing vehicle. By identifying the word 2023/2024 on the sticker, it determines whether the vehicle is registered or not.

Previous methods of managing driving records, such as manual checks and roadblocks, have proven time-consuming and ineffective. The project aims to automate this process, providing a more efficient and scalable solution. Using a dataset containing images of various vehicles and their registration stickers, the team developed a system to track vehicle registration requirements in real time.

The main goal of the project is to develop an automatic registration detection system that can accurately identify registered vehicles in real time. Integrating image recognition and artificial intelligence technologies, the system aims to improve campus security and promote widespread compliance with vehicle registration guidelines.

For the approach, the system checks three types of registration stickers - for students living in UMPSA hostels, students living outside UMPSA hostels and staff for the text '2023/2024'. This verification process involves optical character recognition (OCR) and CV2 technologies. By ensuring sticker control, the system provides a "vehicle is registered" and “vehicle is not registered” message, improving its reliability and efficiency.Additionally, Haar Cascades is employed in the system's code for object detection. Haar Cascades provides a practical solution for vehicle identification from images or live streaming videos, which is an essential part of the vehicle registration sticker recognition system pipeline, prior to further processing such as optical character recognition (OCR) and CV2 techniques to check the registered stickers.

2.2 Objectives

The objectives of the project are:

1. To enhance campus security and ensure a safe atmosphere by employing cutting-edge technology to detect vehicle registration stickers, thereby promoting adherence to vehicle registration guidelines.
2. To analyze registration stickers and distinguish between registered and unregistered vehicles.
3. To enable detection of vehicle registration in different scenarios, including photos and real-time streaming at the university's entry and exit points.

3.0 METHODOLOGY

3.1 Data Collection and Preparation

To adapt the provided text to fit our project on vehicle registration sticker detection, we possess the capability to modify the text accordingly. Our team has diligently gathered and organized image and video datasets to develop an accurate and reliable automatic vehicle registration sticker recognition system.

In order to construct a comprehensive dataset, we actively captured a wide range of real-world scenarios and situations. This meticulous approach ensures that the system can handle diverse data types and offer real-time analytical capabilities. The dataset consists of both static images and dynamic videos, providing a holistic representation of registration sticker usage in different contexts.

The images within the dataset depict various scenes with vehicles, some displaying registration stickers while others do not. This variation accurately reflects the diverse usage of registration stickers in different scenarios. For instance, Image 1 showcases vehicles that have visible registration stickers, as well as vehicles without stickers. Similarly, Image 2 and Image 3 respectively depict vehicles without stickers and vehicles with stickers. Furthermore, Image 4 portrays a scenario where all vehicles in the scene have registration stickers.

In addition to images, we have also recorded videos to simulate real-world scenarios. One video captures a busy street scene with vehicles displaying registration stickers, representing a compliant scenario. Another video showcases a situation where vehicles either have expired stickers or no stickers at all, illustrating non-compliance with registration regulations. These videos enable the system to analyze registration sticker usage in dynamic environments and test its ability to detect stickers in real-time video feeds.

To enhance the system's functionality, we have developed a real-time streaming system. This system allows for the monitoring of registration sticker compliance in real time, enabling us to capture the current situation as a sample video as it unfolds. This video stream provides invaluable data for training and testing the system's real-time video analysis capabilities.

The collected data, including both images and videos, undergoes meticulous preparation and is utilized to train registration sticker detection algorithms. Annotation and other techniques are employed to ensure the accuracy and effectiveness of the algorithms.

3.2 Project Codes

3.2.1 Codes for Live Streaming Sticker Detection

```
import cv2
import pytesseract

# Set the path to the Tesseract-OCR executable
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

# Load the car plate cascade
plate_cascade =
cv2.CascadeClassifier('C:\\Users\\user\\OneDrive\\Documents\\UMPeducation\\Semester4\\Artifi
cial Intelligence\\haarcascade_russian_plate_number.xml')

if plate_cascade.empty():
    print("Error: Failed to load cascade classifier XML file.")
    exit()

def detect_and_recognize_plate(frame):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    plates = plate_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30))

    detected_2023_2024 = False
    label = "Vehicle is not registered"
    color = (0, 0, 255)

    for (x, y, w, h) in plates:
        plate = frame[y:y+h, x:x+w]
        text = pytesseract.image_to_string(plate, config='--psm 8')  # PSM 8 assumes a single
word

        if "2023/2024" in text:
            detected_2023_2024 = True
            print(f"Detected sticker text: {text.strip()}")
            label = "Vehicle is registered"
            color = (0, 255, 0)
            # Only draw one frame and break after detecting "2023/2024"
            cv2.rectangle(frame, (x, y), (x+w, y+h), color, 2)
            cv2.putText(frame, label, (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color, 2)
            break
```

```
    if not detected_2023_2024:
        for (x, y, w, h) in plates:
            # Draw only one frame for non-2023/2024 detections
            cv2.rectangle(frame, (x, y), (x+w, y+h), color, 2)
            cv2.putText(frame, label, (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color, 2)
            break

    return frame

def process_camera_feed():
    cap = cv2.VideoCapture(0)
    if not cap.isOpened():
        print("Error: Could not open camera.")
        return

    while True:
        ret, frame = cap.read()
        if not ret:
            print("Error: Failed to capture image.")
            break

        result_frame = detect_and_recognize_plate(frame)
        cv2.imshow("Camera Feed", result_frame)

        key = cv2.waitKey(1) & 0xFF
        if key == ord('q'):
            break

    cap.release()
    cv2.destroyAllWindows()

if __name__ == "__main__":
    process_camera_feed()
```

3.2.2 Codes for Image Sticker Detection

```
import cv2
import pytesseract
import re

# Set the path to the Tesseract-OCR executable
pytesseract.pytesseract.tesseract_cmd = r'C:\Program
Files\Tesseract-OCR\tesseract.exe'

# Load the car plate cascade
plate_cascade =
cv2.CascadeClassifier('C:\\Users\\user\\OneDrive\\Documents\\UMPeducation\\Semeste
r4\\Artificial Intelligence\\haarcascade_russian_plate_number.xml')

if plate_cascade.empty():
    print("Error: Failed to load cascade classifier XML file.")
    exit()

import cv2
import pytesseract
import re

def detect_and_recognize_plate(frame):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    plates = plate_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30))

    for (x, y, w, h) in plates:
        plate = frame[y:y+h, x:x+w]
        text = pytesseract.image_to_string(plate, config='--psm 8')  # PSM 8 assumes a
single word

        # Regular expression to match "YYYY/YYYY"
        match = re.search(r'\b\d{4}/\d{4}\b', text)

        if match:
            detected_text = match.group()
            if "2023/2024" in detected_text:
                color = (0, 255, 0)
                label = "Vehicle is registered"
            else:
                color = (0, 0, 255)
```

```
            label = "Vehicle is not registered"

        cv2.rectangle(frame, (x, y), (x+w, y+h), color, 2)
        cv2.putText(frame, label, (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color,
        print(f"Detected plate text: {detected_text}")

    return frame


def process_camera_feed():
    cap = cv2.VideoCapture(0)
    if not cap.isOpened():
        print("Error: Could not open camera.")
        return

    while True:
        ret, frame = cap.read()
        if not ret:
            print("Error: Failed to capture image.")
            break

        result_frame = detect_and_recognize_plate(frame)
        cv2.imshow("Camera Feed", result_frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cap.release()
    cv2.destroyAllWindows()

def process_img_feed():
    img = cv2.imread('non registered.png')
    result_frame = detect_and_recognize_plate(img)
    cv2.imshow("Image Feed", result_frame)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

if __name__ == "__main__":
    process_img_feed()
```

3.2.3 Adding GUI to Codes

```
import cv2
import pytesseract
import tkinter as tk
from tkinter import filedialog
from PIL import Image, ImageTk
import threading

# Set the path to the Tesseract-OCR executable
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

# Define the path to the haarcascade XML file for plate detection
cascade_path = 'C:\\Users\\user\\OneDrive\\Documents\\UMPeducation\\Semester4\\Artificial
Intelligence\\haarcascade_russian_plate_number.xml'
plate_cascade = cv2.CascadeClassifier(cascade_path)  # Load the cascade file

# Check if the cascade classifier loaded correctly
if plate_cascade.empty():
    print("Error: Failed to load cascade classifier XML file.")
    exit()

# Function to detect and recognize license plates in a given frame
def detect_and_recognize_plate(frame):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)  # Convert the frame to grayscale
    plates = plate_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30))  # Detect plates

    detected_2023_2024 = False
    label = "Vehicle is not registered"
    color = (0, 0, 255)

    for (x, y, w, h) in plates:
        plate = frame[y:y+h, x:x+w]
        text = pytesseract.image_to_string(plate, config='--psm 8')  # PSM 8 assumes a single
word

        if "2023/2024" in text:
            detected_2023_2024 = True
            print(f"Detected sticker text: {text.strip()}")
            label = "Vehicle is registered"
            color = (0, 255, 0)
            # Only draw one frame and break after detecting "2023/2024"
            cv2.rectangle(frame, (x, y), (x+w, y+h), color, 2)
```

```
            cv2.putText(frame, label, (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color, 2)
            break

    if not detected_2023_2024:
        for (x, y, w, h) in plates:
            # Draw only one frame for non-2023/2024 detections
            cv2.rectangle(frame, (x, y), (x+w, y+h), color, 2)
            cv2.putText(frame, label, (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color, 2)
            break

    return frame

# Function to process the live camera feed
def process_camera_feed():
    cap = cv2.VideoCapture(0)  # Open the default camera
    if not cap.isOpened():
        print("Error: Could not open camera.")
        return

    while True:
        ret, frame = cap.read()  # Read a frame from the camera
        if not ret:
            print("Error: Failed to capture image.")
            break

        result_frame = detect_and_recognize_plate(frame)  # Detect and recognize plates in the
frame
        cv2.imshow("Camera Feed", result_frame)  # Display the processed frame

        if cv2.waitKey(1) & 0xFF == ord('q'):  # Exit the loop if 'q' is pressed
            break

    cap.release()  # Release the camera
    cv2.destroyAllWindows()  # Close all OpenCV windows

# Function to detect sticker from a photo
def detect_sticker_from_photo():
    file_path = filedialog.askopenfilename()  # Open a file dialog to select an image
    if file_path:
        image = cv2.imread(file_path)  # Read the selected image
        if image is None:
            print("Error: Could not read image.")
            return
```

```
        result_frame = detect_and_recognize_plate(image)  # Detect and recognize plates in the image
        cv2.imshow('Sticker Detection', result_frame)  # Display the processed image
        cv2.waitKey(0)  # Wait for a key press
        cv2.destroyAllWindows()  # Close all OpenCV windows

# Function to create the GUI
def create_gui():
    root = tk.Tk()  # Create the main window
    root.title("Sticker Detection GUI")  # Set the window title

    # Load and set background image
    try:
        background_image = Image.open(r"gui pic.jpeg")
        background_photo = ImageTk.PhotoImage(background_image)  # Convert image for Tkinter
    except Exception as e:
        print(f"Error loading background image: {e}")
        return

    background_label = tk.Label(root, image=background_photo)  # Create a label to hold the background image
    background_label.image = background_photo  # Keep a reference to avoid garbage collection
    background_label.place(relwidth=1, relheight=1)  # Place the label to cover the entire window

    # Create title label
    title_label = tk.Label(root, text="UMPSA VEHICLE REGISTRATION STICKER DETECTION", bg='#80c1ff', font=("Helvetica", 16, "bold"))
    title_label.place(relx=0.5, rely=0.05, anchor='n')  # Place the title label

    # Create a frame for the buttons
    button_frame = tk.Frame(root, bg='#80c1ff', bd=5)
    button_frame.place(relx=0.5, rely=0.15, relwidth=0.75, relheight=0.1, anchor='n')  # Place the button frame

    # Create the first button
    button1 = tk.Button(button_frame, text="Sticker Detection from Photo", command=detect_sticker_from_photo)
    button1.place(relwidth=0.5, relheight=1)  # Place the first button

    # Create the second button
    button2 = tk.Button(button_frame, text="Sticker Detection Live Stream", command=process_camera_feed)
```

```
    button2.place(relx=0.5, relwidth=0.5, relheight=1)  # Place the second button

    # Create the close button
    close_button = tk.Button(root, text="Close", command=root.quit)
    close_button.pack(side=tk.BOTTOM, pady=20)  # Place the close button at the bottom

    root.mainloop()  # Start the Tkinter event loop

# Function to run the GUI in a separate thread
def run_gui():
    create_gui()

# Run the GUI in a separate thread to avoid blocking the Jupyter Notebook
gui_thread = threading.Thread(target=run_gui)  # Create a new thread for the GUI
gui_thread.start()  # Start the GUI thread
```

4.0 RESULTS AND DISCUSSION

GUI Output:

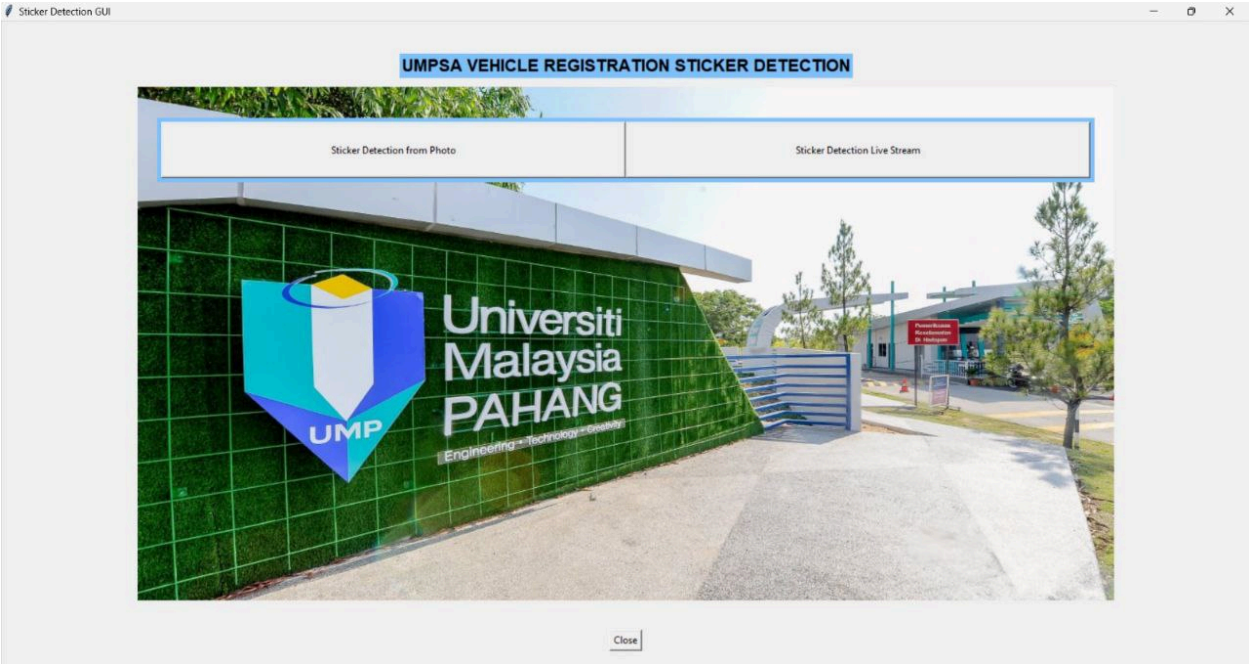

Figure 1: *GUI Output*

When the code is initially started, the user sees the GUI main menu titled UMPSA vehicle registration sticker detection. The user can select two buttons in this main menu: sticker detection from photo and live stream.

Sticker Detection from Photo Output:

Figure 2: *Vehicle of Siswa Residensi sticker is registered.*

When the user clicks the sticker detection from the photo, the file directory opens, and the user may select which picture to detect. In this picture, the student's sticker, who is a Siswa resident, was uploaded, and it was detected that the sticker was registered.

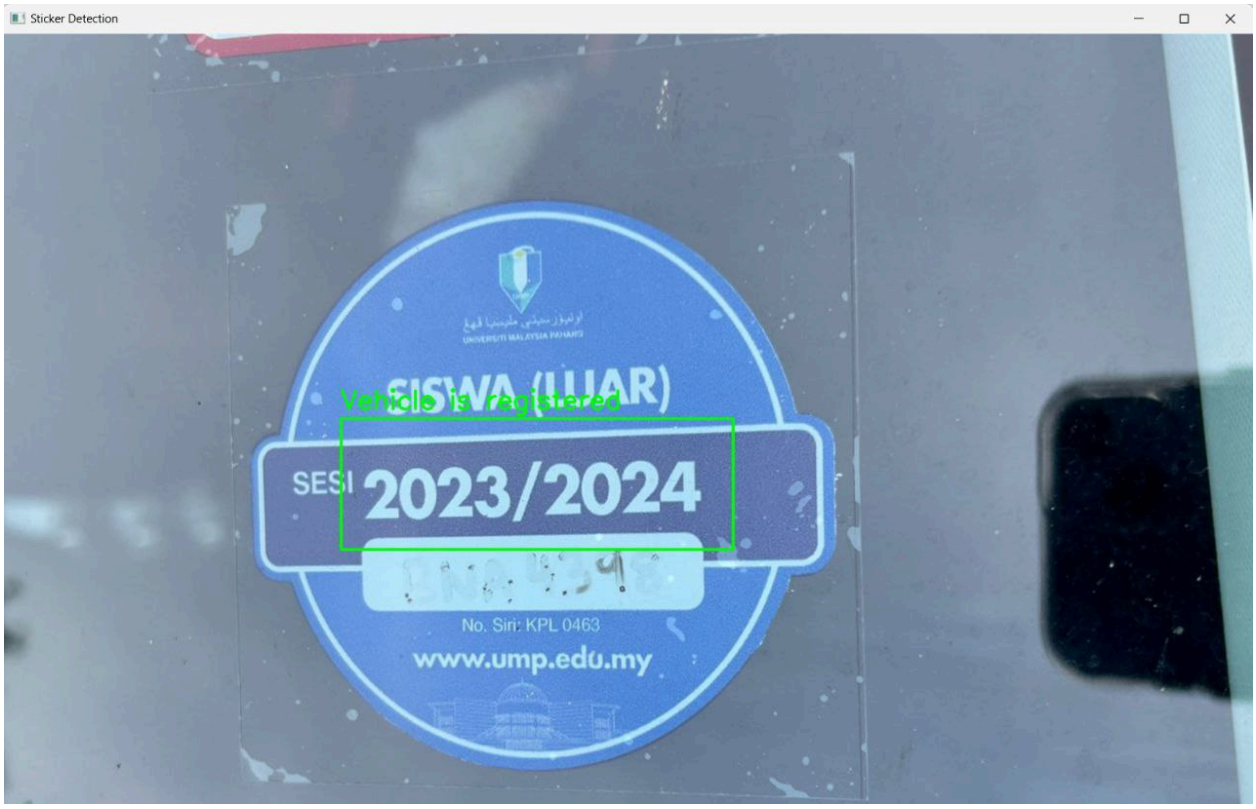

Figure 3: *Vehicle of Siswa Luar sticker is registered*

When the user clicks the sticker detection from the photo, the file directory opens, and the user may select which picture to detect. In this picture, the student's sticker, a Siswa but not a resident, was uploaded, and it was detected that the sticker was registered.

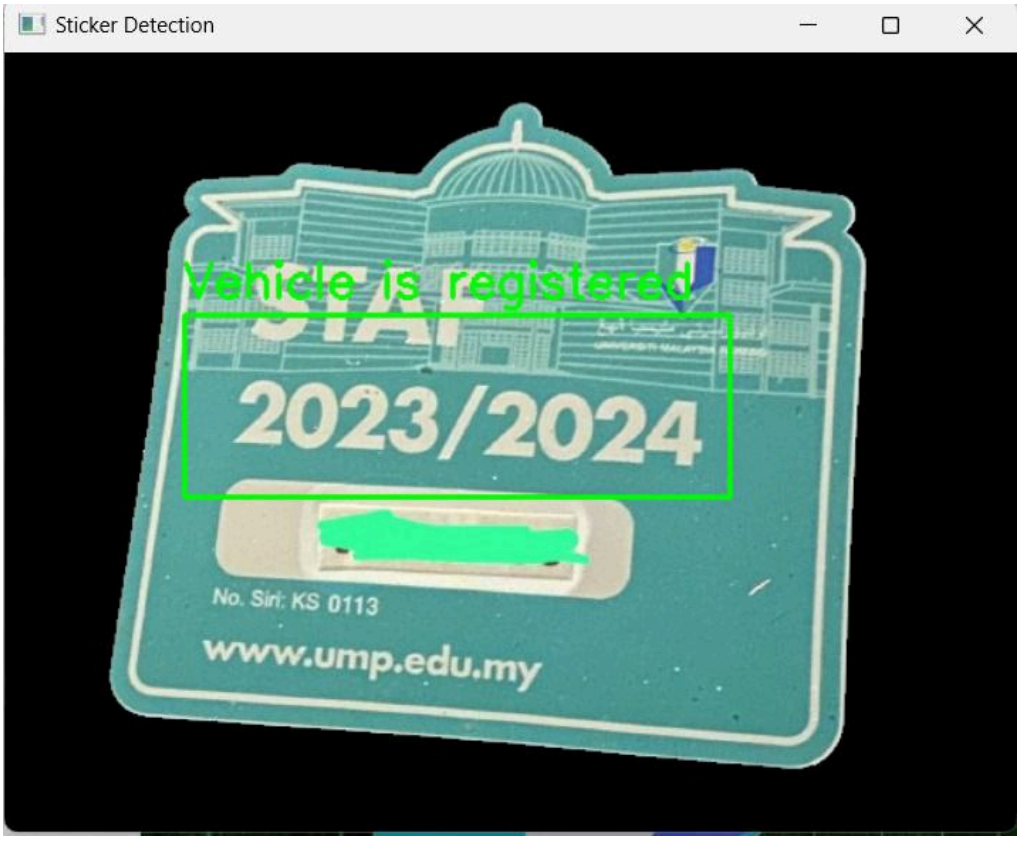

Figure 4: *Vehicle of Staff sticker is registered*

When the user clicks the sticker detection from the photo, the file directory opens, and the user may select which picture to detect. In this picture, the staff's sticker was uploaded, and it was detected that the sticker was registered.

Sticker Detection from Live Streaming Output:

Figure 5: *Student's vehicle sticker is registered*

From the figure above, user shows a student's vehicle sticker. It resulted on "vehicle is registered". Means that the sticker is officially registered for the year. Our system detect only the number 2023/2024 (latest year) on it.

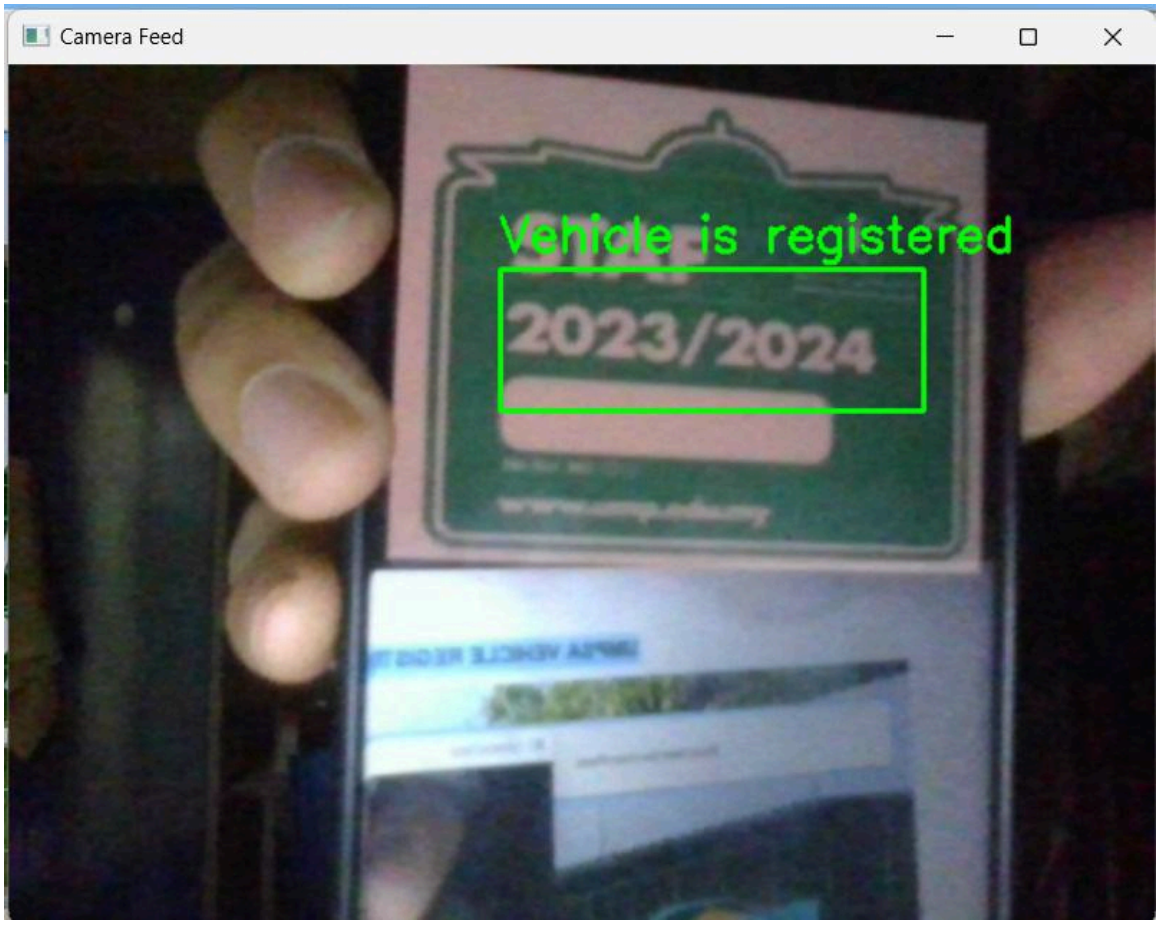

Figure 6: *Staff's vehicle sticker is registered*

From the figure above, the user shows a staff's vehicle sticker. It resulted in "vehicle is registered". Means that the sticker is officially registered for the year. Our system detects only the number 2023/2024 (latest year) on it.

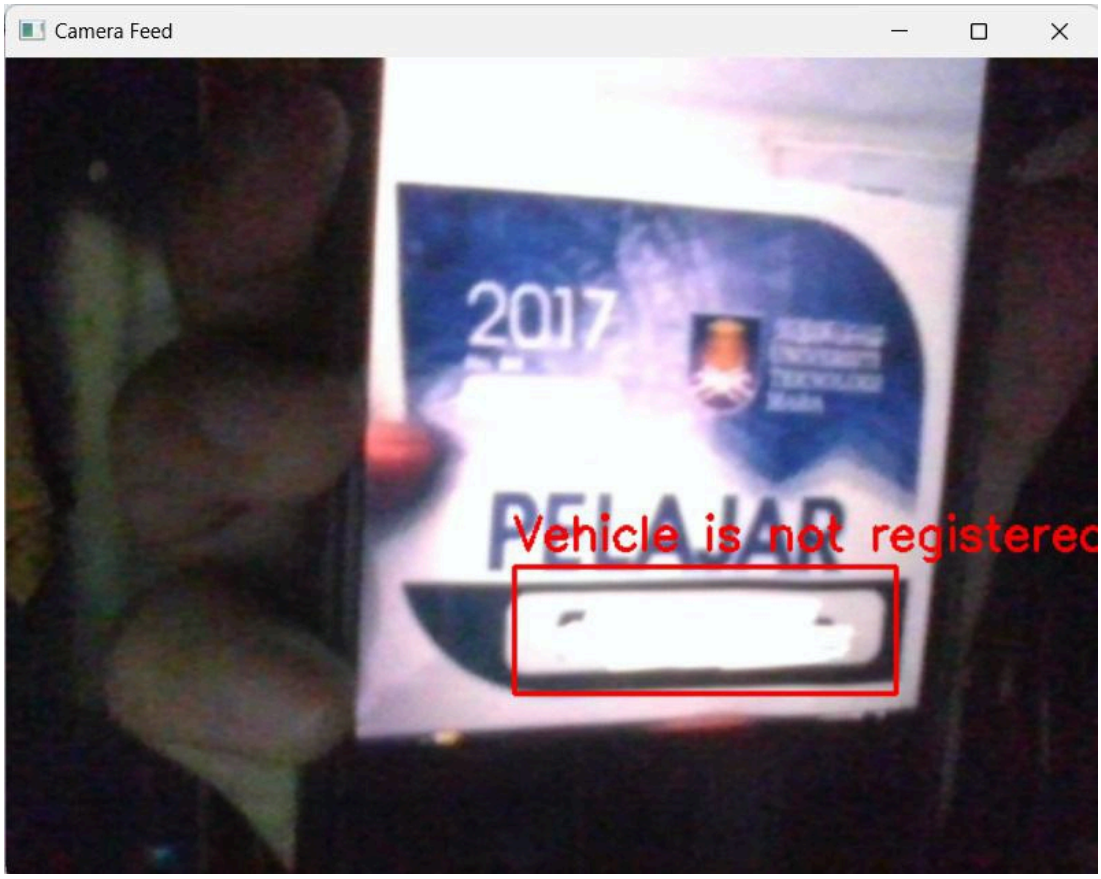

Figure 4: *Student's vehicle sticker is not registered*

From the figure above, the user shows a student's vehicle sticker. It resulted on "vehicle is not registered". Means that the sticker is not officially registered for the year. The sticker showed the year "2017", hence it isn't registered by the system. Our system detects only the number 2023/2024 (latest year) on it.

More outputs are available by clicking on the Google Drive folder and attempting to run the coding with it. All of output can be observed here : Output Data

5.0 LIMITATION

In the current situation, there are problems with the recognition of the entire sticker since the recognition is currently obstructed due to certain restrictions imposed by the Umpsa security organization. These limitations do not allow it to work with the real sticker for training other models or, at least, for some manipulations with the sticker. Therefore, it is impossible to fine tune a detection model that is a hundred percent accurate within current constraints. But they continue to look for other potential opportunities in the future. If ever permitted by the UMPSA security organization to inspect the real sticker, the next step towards training of the model could commence. It would also be a great improvement in our capacity of providing a better way of identifying the sticker better and more accurately. This will be made possible by the actual sticker for training of the machine so as to make it more accurate which helps in enhancing the detectors performance.

6.0 CONCLUSION

The UMPSA Vehicle Registration Sticker Detection project successfully developed a system that uses advanced image processing and Optical Character Recognition (OCR) techniques to detect vehicle registration stickers automatically. This effort aims to increase the accuracy and efficiency of car registration verification, which is critical for traffic management and law enforcement operations.

Despite problems such as sticker appearance variations and environmental influences, the system proved resilient and scalable. It handled multiple designs and conditions well, responding to various lighting, weather, and vehicle speed scenarios using robust pre-processing and adaptive algorithms.

Future development will concentrate on ongoing improvement by collecting new data, developing the system to recognize additional types of vehicle documents, and incorporating user feedback from law enforcement and traffic management staff. This study exhibits AI's potential for automating car registration verification and paves the way for future breakthroughs in traffic control systems.

7.0 REFERENCES

Tesseract-Ocr. (n.d.). *GitHub - tesseract-ocr/tesseract: Tesseract Open Source OCR Engine (main repository)*. GitHub. https://github.com/tesseract-ocr/tesseract

How to install easy OCR. (n.d.). Stack Overflow. https://stackoverflow.com/questions/65630824/how-to-install-easy-ocr

Lihini. (n.d.). *GitHub - lihini223/Object-Detection-model: Simple Object Detection Model in Python*. GitHub. https://github.com/lihini223/Object-Detection-model

GeeksforGeeks. (2023b, May 16). *Opencv Python program for Face Detection*. GeeksforGeeks. https://www.geeksforgeeks.org/opencv-python-program-face-detection/

Nurfikri, F. (2022, November 2). *How to build Optical Character Recognition (OCR) in Python*. Built In. https://builtin.com/data-science/python-ocr

Maleehak. (n.d.). *GitHub - Maleehak/Car-number-plate-recognition-using-OpenCV: Recognize cars using Haar cascade and OpenCV*. GitHub. https://github.com/Maleehak/Car-number-plate-recognition-using-OpenCV

GeeksforGeeks. (2024b, March 20). *Detect and Recognize Car License Plate from a video in real time*. GeeksforGeeks. https://www.geeksforgeeks.org/detect-and-recognize-car-license-plate-from-a-video-in-real-time/

Python, R. (2022, April 1). *PySimpleGUI: The simple way to create a GUI with Python*. https://realpython.com/pysimplegui-python/