LICENSE PLATE RECOGNISATION SYSTEM

Batch 33:
Batch Members:
NAGIREDDY RAMESH KUMAR REDDY (22781A04B2)
MUTTINENI NITHIN (22781A04B0)
MUKTHAPURAM SAI KEERTHAN REDDY (22781A04A8)
PALADAGU PRAVEEN (22781A04C3)

Problem statement:

Develop a real-time License Plate Recognition (LPR) system using a Raspberry Pi and camera module to accurately capture and recognize license plates in various environmental conditions. The system must prioritize real-time processing, high recognition accuracy, and compatibility with Raspberry Pi hardware constraints. This solution aims to support applications such as parking management and security by providing efficient and reliable license plate identification and integration capabilities.

LICENSE PLATE RECOGNISATION SYSTEM

Scope of solution:

The scope of the solution for the real-time License Plate Recognition (LPR) system using Raspberry Pi and Camera includes several key aspects:

Hardware Setup:

Selection and setup of Raspberry Pi hardware (preferably Raspberry Pi 4) and compatible camera module (such as Raspberry Pi Camera Module V2). Ensuring adequate power supply, storage (micro SD card), and optional internet connectivity for updates.

Software Installation and Configuration:

Installation of Raspberry Pi OS and necessary libraries (e.g., Open CV, Open ALPR) for image processing and license plate recognition. Configuration of the camera interface and development environment (Python).

Image Capture and Processing:

LICENSE PLATE RECOGNISATION SYSTEM

Development of Python scripts using Open CV to capture images from the camera module.
Implementation of image preprocessing techniques (e.g., resizing, normalization) to enhance recognition accuracy.
Extraction of license plate regions from captured images using computer vision techniques.
License Plate Recognition (LPR):

Integration of Open ALPR library or similar for accurate license plate recognition.
Fine-tuning recognition parameters to optimize accuracy and performance.
Implementing algorithms to handle variations in lighting, plate orientation, and background noise for robust recognition.
Real-time Processing:

Optimization of image capture and recognition pipeline to achieve real-time performance.
Consideration of threading or multiprocessing techniques to parallelize tasks and improve efficiency.
Integration and Application Development:

LICENSE PLATE RECOGNISATION SYSTEM

Integration of the LPR system with applications such as parking management or security systems.
Development of interfaces (e.g., web-based, command-line) for system monitoring and interaction.
Implementing actions based on recognition results (e.g., opening gates, logging entries).
Testing and Validation:

Comprehensive testing under various environmental conditions (daylight, night, shadows) to ensure reliability.
Performance testing with different vehicle speeds and angles to validate real-time capabilities.
Iterative refinement of algorithms and parameters based on testing results to enhance system accuracy and robustness.
Documentation and Deployment:

Documentation of system architecture, installation steps, and operational guidelines.

LICENSE PLATE RECOGNISATION SYSTEM

Required components to develop solution:

Hardware components:

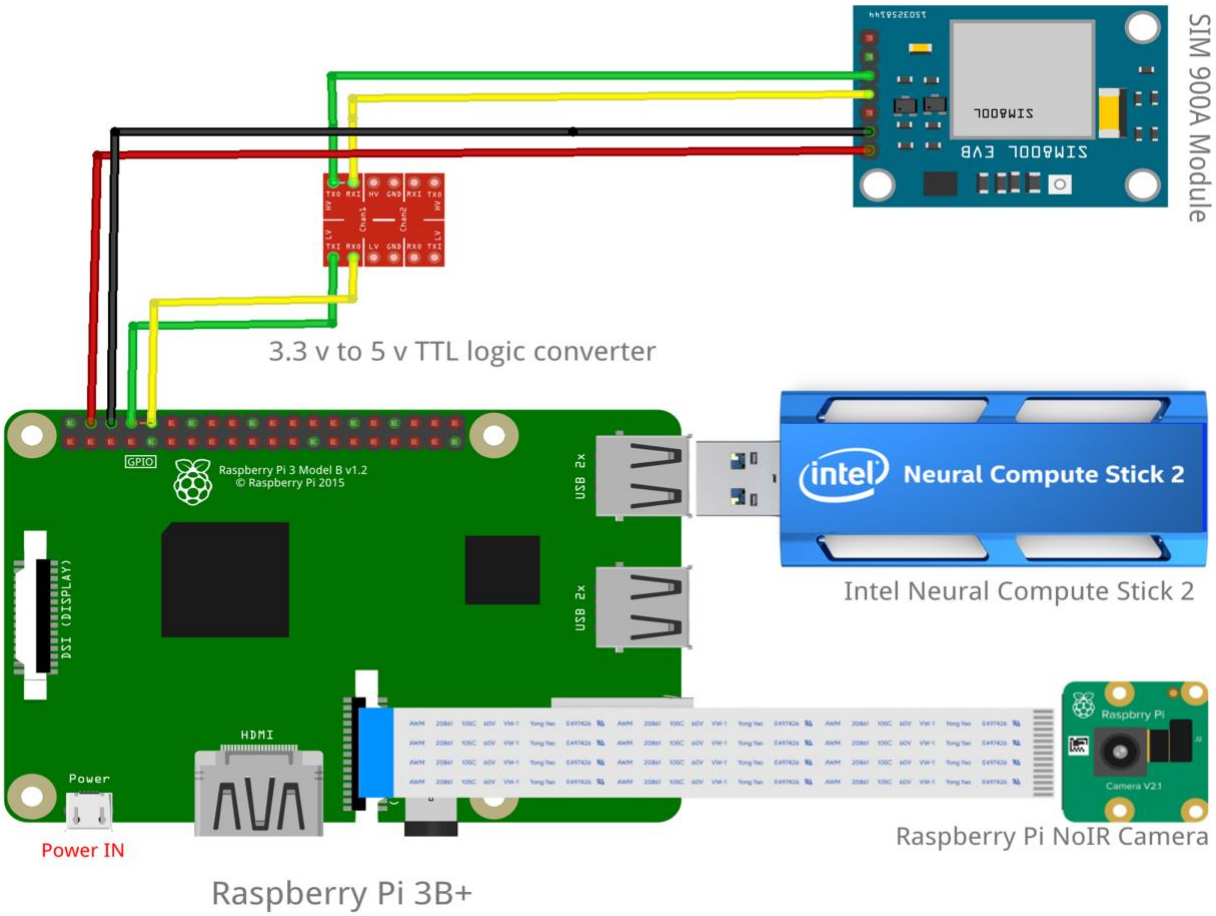
- Camera
- Processing Unit
- SSD
- Wi-Fi

Software components:

- Image Acquisition Software
- Image Processing Algorithm
- License Plate Detector
- Optical Character Recognition(OCR)System
- Database Integration
- User Interface
- Machine Learning

LICENSE PLATE RECOGNISATION SYSTEM

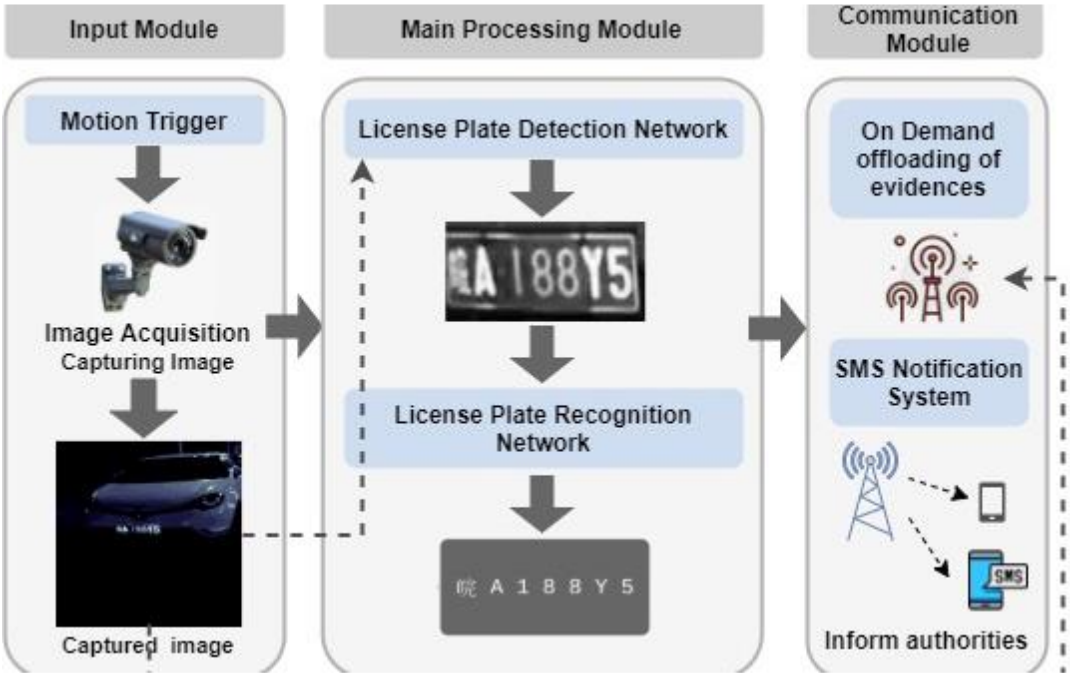
Simulated Circuit:



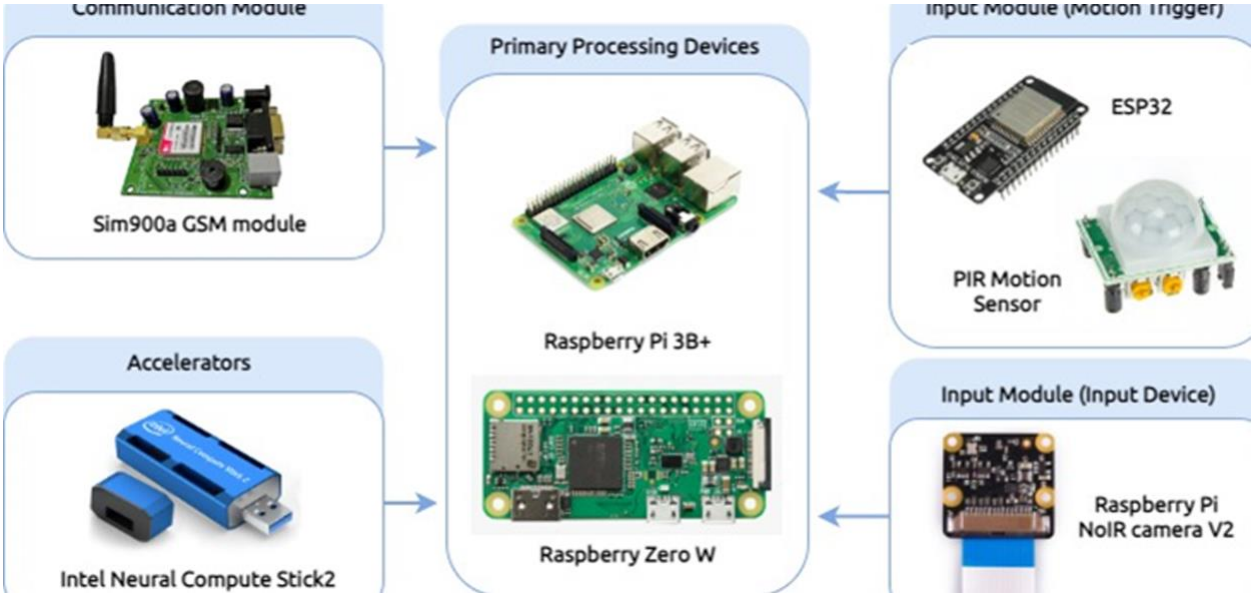
LICENSE PLATE RECOGNISATION SYSTEM

THE PROCESS OF RECOGNITION:

1. Overview of the proposed model:

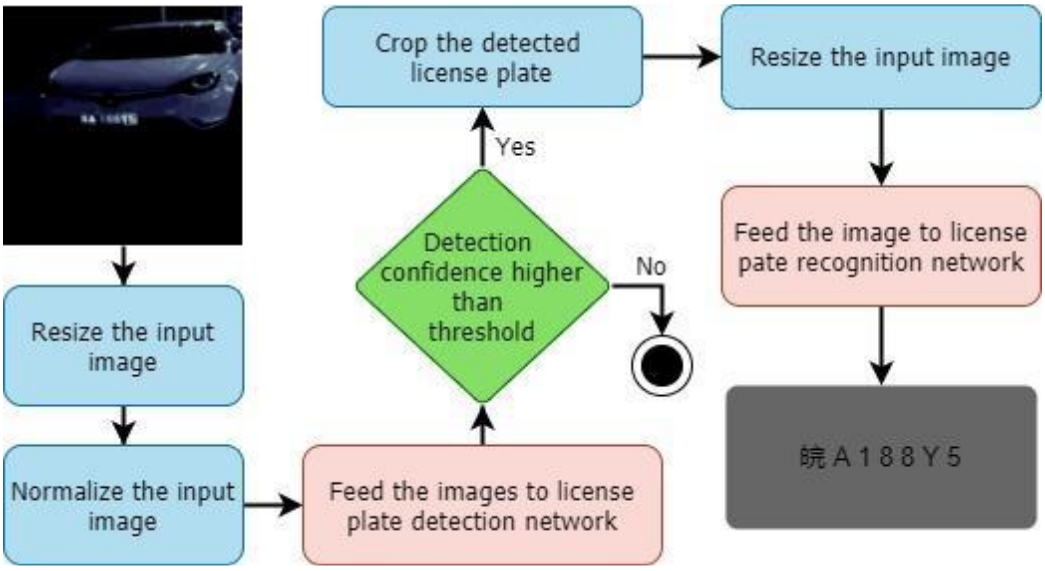


2. Hardware stack of the proposed solution:

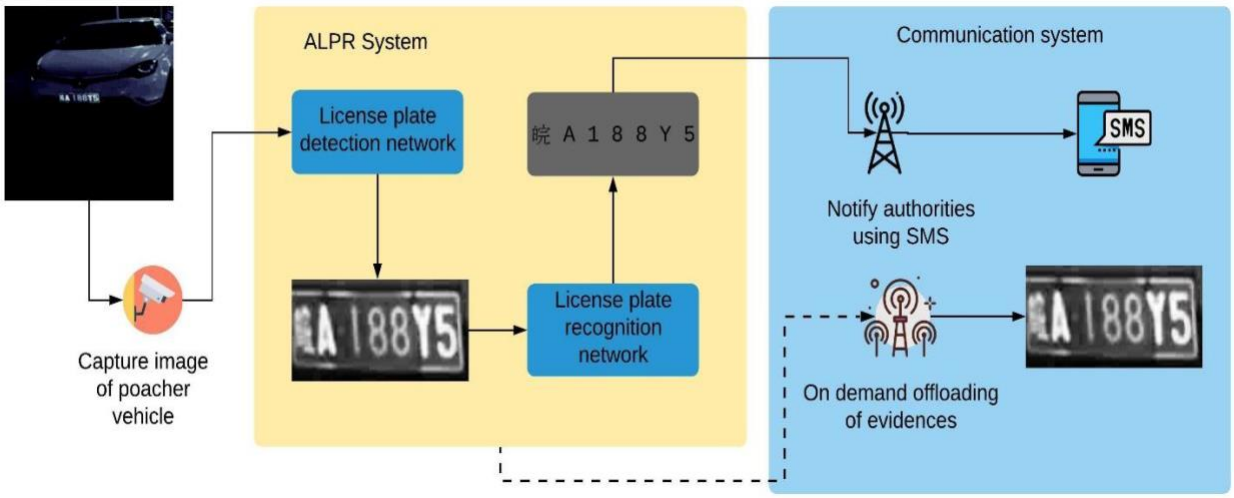


LICENSE PLATE RECOGNISATION SYSTEM

3. Two-stage license plate recognition pipeline:



4. Data flow of the proposed system:



LICENSE PLATE RECOGNISATION SYSTEM

Video of the demo:

Code for the solution:

```
import cv2
import numpy as np
import tflite_runtime.interpreter as tflite
import serial
import time

# Initialize the camera
camera = cv2.VideoCapture(0)

# Load TensorFlow Lite model
interpreter =
tflite.Interpreter(model_path="model.tflite")
interpreter.allocate_tensors()

# Define function for license plate recognition
def recognize_license_plate(frame):
    # Preprocess image for model input
```

LICENSE PLATE RECOGNISATION SYSTEM

```
    input_tensor_index =
interpreter.get_input_details()[0]['index']
    output_tensor_index =
interpreter.get_output_details()[0]['index']

    # Resize image to model input size
    resized_frame = cv2.resize(frame, (224, 224))  #
Change size based on your model
    input_data = np.expand_dims(resized_frame,
axis=0).astype(np.float32)

    # Run inference
    interpreter.set_tensor(input_tensor_index,
input_data)
    interpreter.invoke()
    output_data =
interpreter.get_tensor(output_tensor_index)

    # Post-process output to recognize license plate
    # Assuming output_data contains the recognized plate
    license_plate = output_data[0]
    return license_plate

# Initialize SIM900A for sending data
ser = serial.Serial('/dev/serial0', 9600)  # Adjust serial
port if needed
```

LICENSE PLATE RECOGNISATION SYSTEM

```
try:
    while True:
        ret, frame = camera.read()
        if not ret:
            continue

        # Perform license plate recognition
        license_plate = recognize_license_plate(frame)
        print(f"Recognized License Plate: {license_plate}")

        # Send data to SIM900A
        ser.write(f"Recognized License Plate:
{license_plate}\n".encode())
        time.sleep(1)

except KeyboardInterrupt:
    camera.release()
    ser.close()
```