Traffic Scene Generation and Obstacle Detection Using Generative AI and Object Detection

# 1. Project Overview

This project demonstrates a simple yet complete pipeline that:  
1. Generates synthetic top-down traffic scenes using a generative AI model (Stable Diffusion),  
2. Analyzes these scenes to detect surrounding objects and potential obstacles using an object detection model (DETR), and  
3. Makes basic navigation decisions (e.g., avoid or proceed) based on detected objects.  
  
This system simulates a lightweight autonomous driving assistant using publicly available AI models, implemented and tested using Google Colab for GPU acceleration.

# 2. Tools and Technologies Used

- Google Colab: For cloud-based GPU computing  
- Hugging Face Hub: For accessing pre-trained models  
- Diffusers Library: For image generation via Stable Diffusion  
- Transformers Library: For object detection via DETR (DEtection TRansformer)  
- PyTorch: For deep learning computation  
- PIL (Python Imaging Library): For image processing and visualization  
- Matplotlib: For rendering annotated image outputs

# 3. Methodology

## a. Scene Generation

A textual prompt (e.g., "a top-down view of 3 cars driving on a road with buildings and trees around") is passed to Stable Diffusion. The generated image is saved locally. The model was configured to run on GPU to reduce generation time and enhance visual fidelity.

## b. Object Detection

The generated image is processed using facebook/detr-resnet-50. DETR identifies objects in the scene such as cars, buses, pedestrians, trees, and traffic signs. A detection threshold is applied to remove low-confidence detections.

## c. Navigation Decision

Detected objects are compared against a set of known obstacles. Based on this, the system prints either a warning to avoid (e.g., "Turn left or right") or a go-ahead signal ("Path is clear").

## d. Visual Results

Detection results are annotated and saved on top of the original image. Each object is marked with bounding boxes and labels for easier visual inspection.

# 4. Improvements Made

- Visual Annotation: Added bounding box visualization for detected objects using Matplotlib.  
- GPU Enforcement: Modified code to force usage of GPU if available in Colab.  
- Improved Navigation Logic: Upgraded obstacle-checking system for clarity and accuracy.

# 5. Results and Observations

The model successfully generates traffic scenes and identifies common obstacles like vehicles, pedestrians, buildings, etc. Navigation decisions are reasonable and correspond to the detection output. Visual feedback in the form of annotated images is provided after each detection process.

# 6. Limitations and Future Work

## Performance

The current pipeline is time-consuming:  
- On CPU: ~44 minutes end-to-end  
- On Colab GPU: ~30 minutes  
The DETR model is relatively heavy for real-time use and may not scale well in large simulations.

## Image Quality

The generated scenes often lack realism and spatial coherence. Objects can be malformed or blurry, reducing detection accuracy.

## Future Directions

- Fine-tune the Stable Diffusion model with domain-specific traffic data for improved generation.  
- Use more accurate or lightweight detectors like YOLOv8 or EfficientDet.  
- Introduce semantic segmentation for better spatial understanding.  
- Explore reinforcement learning for dynamic navigation decisions based on real-time detections.