FocusRide: Navigating Attention for Safer Journeys

By **Group 8: Riyanshi Bohra ©,**

Shashwat Singh,

Veeramani Pugazhenthi,

Ajay Sreekumar

Course: ECE 479/579 (Principles of Artificial Intelligence)

Introduction

The "FocusRide" project is designed to fundamentally enhance the way we approach road safety and travel efficiency. At the heart of our innovation is the integration of two key technological advancements: distracted driving behaviour monitoring and dynamic route optimization. This system aims not only to identify and mitigate instances of distracted driving through real-time monitoring but also to optimize travel routes based on the detected state of driver attention. By doing so, FocusRide offers a novel solution that ensures both safer and more efficient journeys, addressing critical needs in today's increasingly busy and distraction-filled driving environments.

Importance of AI and Advanced Technologies in Road Safety:

The advent of Artificial Intelligence (AI) and related technologies has ushered in a new era of possibilities in numerous fields, particularly in enhancing road safety. AI algorithms are capable of processing vast amounts of data from various sensors in real-time, enabling them to detect subtle patterns and changes in driver behaviour that signify distraction. By leveraging such capabilities, FocusRide employs sophisticated deep learning models to continuously assess the driver’s focus and alertness.

Moreover, advanced technologies facilitate the intelligent optimization of travel routes. Unlike traditional navigation systems that primarily consider distance and estimated travel times, FocusRide’s route optimization algorithms also take into account the current state of the driver. For instance, if signs of fatigue or significant distraction are detected, the system can alter the route to include fewer complex driving scenarios, such as avoiding high-traffic areas or intricate intersections. This dynamic adjustment not only helps in reducing the risk of accidents but also ensures that the driving conditions are suited to the driver’s current state, thereby enhancing overall travel efficiency.

Together, these AI-driven solutions form a comprehensive system that acts as a virtual co-pilot, steering us towards a future where road safety is significantly improved through technology. As such, FocusRide not only contributes to the field of intelligent transportation systems but also serves as a model for the integration of AI and advanced technologies in addressing real-world challenges.

Problem Statement

Challenges in Transportation:

Today's transportation systems face a dual challenge of ensuring safety while optimizing efficiency. As road networks become increasingly congested, the incidence of distracted driving has emerged as a significant public safety concern. Distracted driving, which includes behaviours such as texting, adjusting the navigation or entertainment systems, and even momentary lapses in attention, has been identified as a leading cause of road accidents globally. These accidents not only result in significant loss of life and injury but also contribute to increased traffic congestion, delays, and the associated economic costs.

Additionally, traditional route optimization technologies often focus solely on reducing travel times without considering the driver's state, which can lead to recommendations that may not be safe under certain conditions. For example, a route that is optimal in terms of time might involve complex driving manoeuvres or travel through congested urban centres, which can be challenging for a distracted or fatigued driver to navigate safely.

Need for Innovative Solutions in Mobility:

The limitations of current transportation safety and efficiency measures highlight the need for innovative solutions that integrate more comprehensive considerations of driver state into route planning and real-time driving assistance. Such solutions must leverage advanced technologies to detect and mitigate risky behaviours by drivers, and simultaneously adapt their travel routes to suit their current levels of alertness and focus.

FocusRide addresses these challenges by using a sophisticated AI-driven system that not only monitors for signs of distracted driving but also dynamically adjusts travel routes in real time based on the driver's observed state. This approach ensures that routes are not only time-efficient but also safe for the specific conditions and capabilities of the driver at any given moment.

Vision for Enhanced Road Safety and Efficiency:

The overarching goal of the FocusRide project is to redefine the safety and efficiency paradigms in modern transportation. By integrating distracted driving behaviour monitoring with dynamic route optimization, FocusRide aims to significantly reduce the incidence of accidents caused by distracted driving. This system not only serves to protect individual drivers but also contributes to broader societal benefits by improving overall traffic flow and reducing the economic losses associated with road accidents and inefficient travel routes. Through FocusRide, we envision a future where technology and mobility converge to create safer, smarter, and more responsive transportation systems.

System Requirements

Technical Specifications:

1. **Driver Focus Monitoring:**

Sensors: The system will utilize high-resolution cameras and infrared sensors to monitor the driver's eye movements, head position, and facial expressions to detect signs of distraction or fatigue.

Data Processing: Utilize real-time data processing with a deep learning model trained to recognize patterns indicative of distracted driving. The system must process data with a latency of no more than two seconds to ensure timely responses.

Alert System: Integration of a multi-modal alert system that uses audio, visual, and haptic feedback to alert the driver upon detection of any distraction-related behaviour.

2. **Real-Time Route Adjustments:**

Dynamic Navigation: The system will incorporate GPS technology with real-time traffic data to dynamically adjust routes. It must be capable of recalculating routes within five seconds of detecting a change in driver focus.

Safety-Optimized Routing: Algorithms must prioritize routes that require less complex driving manoeuvres or have lower traffic densities when signs of driver distraction or fatigue are detected.

Environmental Adaptability: The system should be capable of integrating weather and road condition data to suggest the safest routes under varying environmental conditions.

3. **Accessibility and Usability Requirements:**
    1. **Interface Accessibility:**
        a. **Visual Design:** High-contrast displays and large fonts for easy reading by drivers with visual impairments.
        b. **Voice Commands:** Comprehensive voice command capabilities to allow hands-free operation and ensure accessibility for drivers with physical disabilities affecting manual interaction.
        c. **Adjustable Alerts:** The alert system should be customizable, allowing drivers to choose between different types of alerts (audio, visual, haptic) based on personal preference and specific disabilities.

    2. **Ease of Use:**
        a. **User-Friendly Interface:** The navigation and alert system interfaces should be intuitive, with minimal learning curve to accommodate users of all ages and technical skills.
        b. **Customization Options:** Provide options for drivers to set their preferences for route options, such as avoiding highways or selecting the

simplest route, and adjust the sensitivity of the distraction detection system.

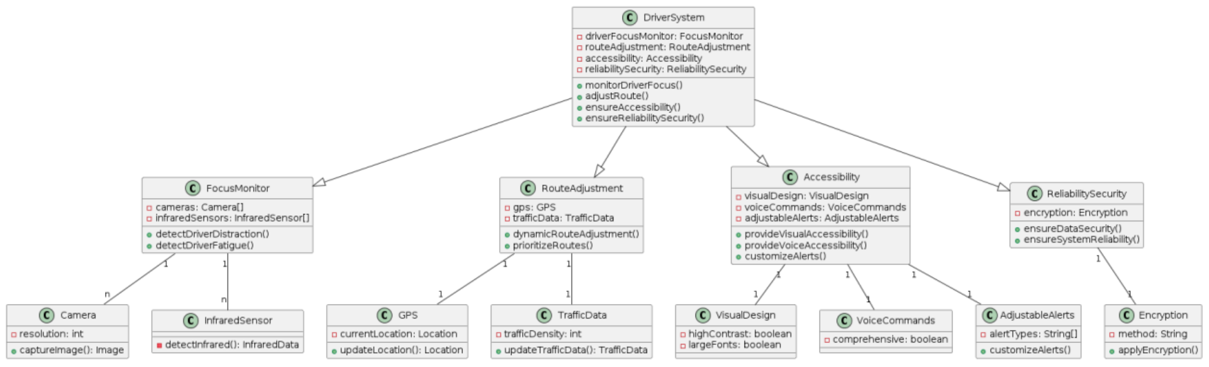
3. **Inclusive Design:**
    a. **Multi-Lingual Support:** The system should support multiple languages to cater to a diverse user base.
    b. **Ergonomic Considerations:** Ensure that all physical components of the system, such as cameras and sensors, do not obstruct the driver's view or interfere with driving operations, and can be adjusted to suit different vehicle interiors.

4. **Reliability and Security Requirements:**

    a. **Data Security:** Implement robust encryption methods for transmitting and storing data collected from drivers to protect their privacy.
    b. **System Reliability:** The system must operate with at least 99.5% uptime and include fail-safe measures to switch to manual controls smoothly in case of a system failure.

System Architecture

UML diagrams or similar modelling specifications to outline the architecture.



This UML diagram outlines the architecture of the FocusRide System:

Driver System: This class represents the main component of the system. It encapsulates various subsystems responsible for different functionalities such as monitoring driver focus, adjusting routes, ensuring accessibility, and ensuring reliability/security.

Focus Monitor: This subsystem is responsible for monitoring the driver's focus. It utilizes cameras and infrared sensors to detect distractions and fatigue.

Route Adjustment: This subsystem handles route adjustment based on GPS and traffic data. It can dynamically adjust routes and prioritize them based on various factors.

Accessibility: This subsystem ensures that the system is accessible to all users. It includes features such as visual accessibility (high contrast, large fonts), voice commands, and customizable alerts.

Reliability Security: This subsystem focuses on ensuring the reliability and security of the system. It includes encryption methods for data security and mechanisms for ensuring system reliability.

Camera and Infrared Sensor: These are components used by the Focus Monitor subsystem to detect distractions and fatigue in the driver.

GPS and Traffic Data: These components are used by the Route Adjustment subsystem to update the current location and traffic data, respectively.

Visual Design, Voice Commands, and Adjustable Alerts: These are components used by the Accessibility subsystem to provide visual accessibility, support voice commands, and customize alerts, respectively.

Encryption: This component is used by the Reliability Security subsystem to apply encryption methods for securing data.

The associations between these classes depict how they are related to each other. For example, each Driver System contains one instance of each of the subsystems (Focus Monitor, Route Adjustment, Accessibility, Reliability Security), and each subsystem contains the necessary components to fulfil its functionalities. Additionally, the relationships between classes such as Focus Monitor with Camera and Infrared Sensor indicate that Focus Monitor uses multiple instances of these components for its operations.

Implementation Details

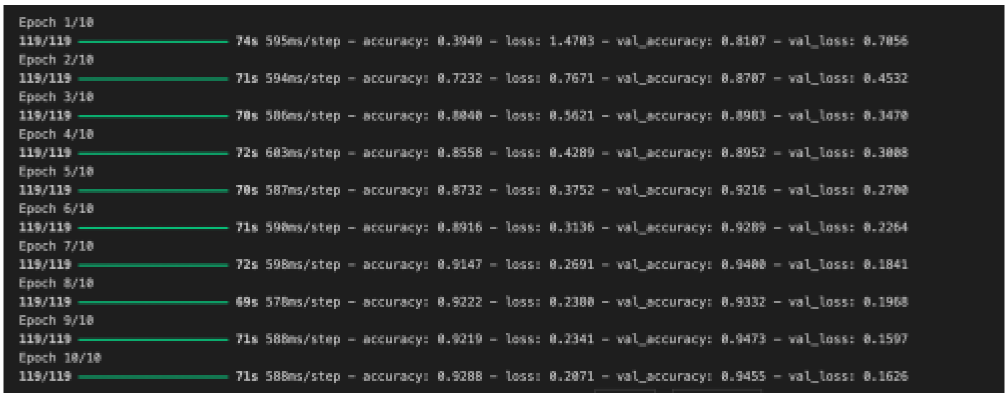
Based on the provided UML diagram, FocusRide’s deep learning model for detecting distracted driving utilizes the components and relationships outlined in the Focus Monitor subsystem:

1. **Input Data:** Images captured by cameras: These images serve as the input data for the deep learning model. They provide visual information about the driver's behaviour.

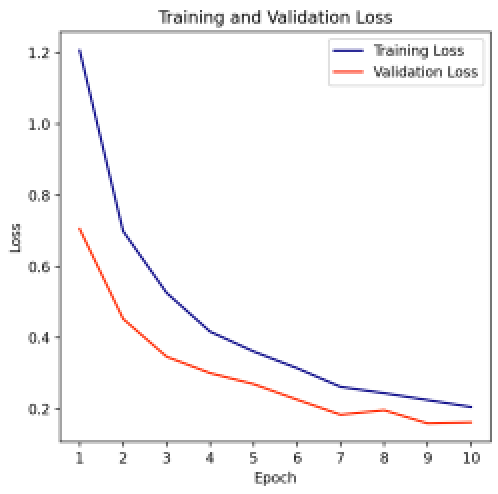
2. **Pre-processing:** The images captured by cameras may need pre-processing steps such as resizing, normalization, and augmentation to prepare them for input into the deep learning model.

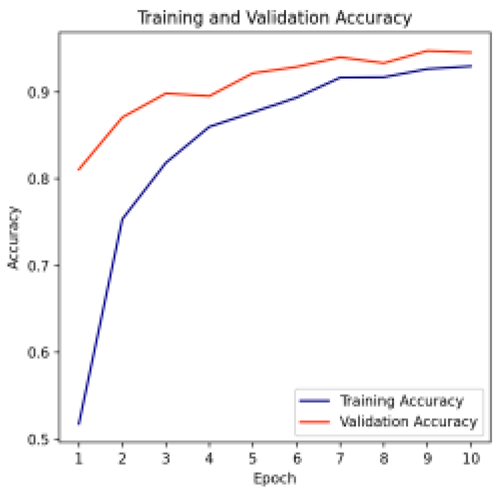
3. **Deep Learning Model:** The deep learning model would be responsible for analysing the input images to detect distracted driving behaviour. It utilizes convolutional neural networks (CNNs) suitable for image classification tasks. The model is trained on a dataset of labelled images, where each image is labelled with the corresponding driving behaviour (e.g., distracted, alert). The model learns to identify patterns and features indicative of distracted driving, such as eyes off the road, head turned away from the windshield, or hands off the steering wheel. For our model, we attempted to use the **‘ResNet50’** and **‘AlexNet’** models first, but ended up choosing **‘MobileNetV2’** model due to its superior performance and metrics over the other two models.

1. MobileNetV2

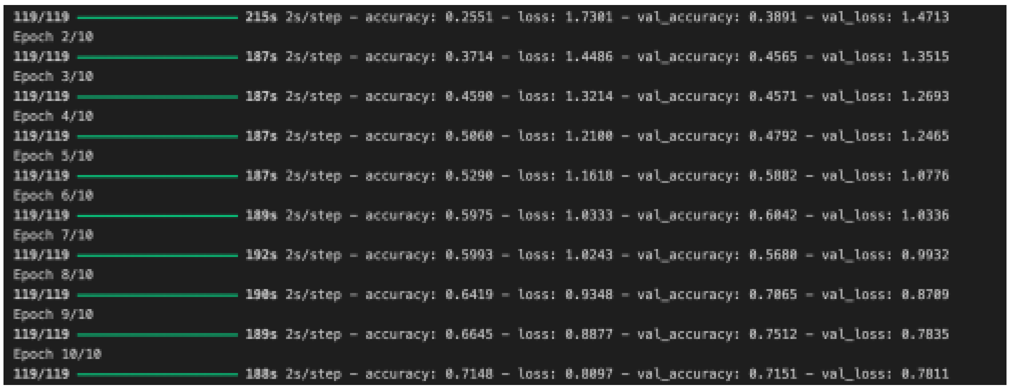


```
Epoch 1/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 74s 595ms/step - accuracy: 0.3949 - loss: 1.4703 - val_accuracy: 0.8107 - val_loss: 0.7056
Epoch 2/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 71s 594ms/step - accuracy: 0.7232 - loss: 0.7671 - val_accuracy: 0.8707 - val_loss: 0.4532
Epoch 3/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 70s 586ms/step - accuracy: 0.8040 - loss: 0.5621 - val_accuracy: 0.8983 - val_loss: 0.3470
Epoch 4/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 72s 603ms/step - accuracy: 0.8558 - loss: 0.4289 - val_accuracy: 0.8952 - val_loss: 0.3008
Epoch 5/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 70s 587ms/step - accuracy: 0.8732 - loss: 0.3752 - val_accuracy: 0.9216 - val_loss: 0.2700
Epoch 6/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 71s 590ms/step - accuracy: 0.8916 - loss: 0.3136 - val_accuracy: 0.9289 - val_loss: 0.2264
Epoch 7/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 72s 598ms/step - accuracy: 0.9147 - loss: 0.2691 - val_accuracy: 0.9400 - val_loss: 0.1841
Epoch 8/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 69s 578ms/step - accuracy: 0.9222 - loss: 0.2380 - val_accuracy: 0.9332 - val_loss: 0.1968
Epoch 9/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 71s 588ms/step - accuracy: 0.9219 - loss: 0.2341 - val_accuracy: 0.9473 - val_loss: 0.1597
Epoch 10/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 71s 588ms/step - accuracy: 0.9288 - loss: 0.2071 - val_accuracy: 0.9455 - val_loss: 0.1626
```

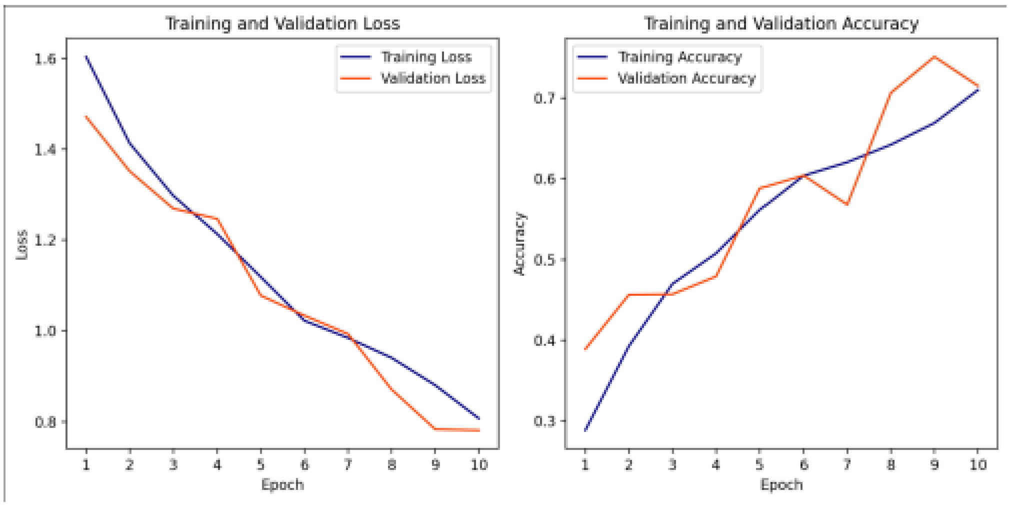




2. ResNet50



```
119/119 ━━━━━━━━━━━━━━━━━━━━ 215s 2s/step - accuracy: 0.2551 - loss: 1.7301 - val_accuracy: 0.3891 - val_loss: 1.4713
Epoch 2/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 187s 2s/step - accuracy: 0.3714 - loss: 1.4486 - val_accuracy: 0.4565 - val_loss: 1.3515
Epoch 3/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 187s 2s/step - accuracy: 0.4590 - loss: 1.3214 - val_accuracy: 0.4571 - val_loss: 1.2693
Epoch 4/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 187s 2s/step - accuracy: 0.5060 - loss: 1.2100 - val_accuracy: 0.4792 - val_loss: 1.2465
Epoch 5/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 187s 2s/step - accuracy: 0.5290 - loss: 1.1610 - val_accuracy: 0.5882 - val_loss: 1.0776
Epoch 6/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 189s 2s/step - accuracy: 0.5975 - loss: 1.0333 - val_accuracy: 0.6042 - val_loss: 1.0336
Epoch 7/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 192s 2s/step - accuracy: 0.5993 - loss: 1.0243 - val_accuracy: 0.5680 - val_loss: 0.9932
Epoch 8/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 190s 2s/step - accuracy: 0.6419 - loss: 0.9348 - val_accuracy: 0.7065 - val_loss: 0.8709
Epoch 9/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 189s 2s/step - accuracy: 0.6645 - loss: 0.8877 - val_accuracy: 0.7512 - val_loss: 0.7835
Epoch 10/10
119/119 ━━━━━━━━━━━━━━━━━━━━ 188s 2s/step - accuracy: 0.7148 - loss: 0.8097 - val_accuracy: 0.7151 - val_loss: 0.7811
```



4. **Output:** The output of the deep learning model will be a prediction or probability score indicating the likelihood of distracted driving behaviour in the input image. This output could be further processed to trigger alerts or interventions if the model detects a high probability of distracted driving.

5. **Integration with Focus Monitor:** The deep learning model would be integrated into the Focus Monitor subsystem, which orchestrates the detection of distracted driving behaviour. The Focus Monitor subsystem would coordinate the capture of images from cameras, pre-processing of images, feeding images into the deep learning model, and interpreting the model's output.

6. **Feedback Loop:** The system may incorporate a feedback loop to continuously improve the deep learning model's performance. Feedback from real-world usage scenarios, including correct and incorrect predictions, can be used to retrain and fine-tune the model over time.

Predicate Logic Conditions:

1. **Class 0 and Class 1 Distraction Levels:** If the driver is not distracted (Class 0) or minimally distracted (Class 1), the system will proceed to find the optimal route using Dijkstra's Algorithm.
2. **Class 2, 3, and 4 Distraction Levels:** If the driver is moderately (Class 2), highly (Class 3), or severely distracted (Class 4), the system will only send out alerts to the driver to stop the vehicle.

Predicate Logic Evaluation: Define predicate logic conditions based on the distraction level reported by the Focus Monitor subsystem.

For example: If distraction_level <= 1: Use Dijkstra's Algorithm to find the optimal route.

If distraction_level > 1: Send out alerts to the driver to stop the vehicle.

Integration with Route Adjustment Subsystem: Within the dynamicRouteAdjustment() method of the Route Adjustment class, implement the predicate logic conditions to determine the course of action. If the distraction level meets the criteria for using Dijkstra's Algorithm, invoke the algorithm to find the optimal route. If the distraction level indicates higher levels of distraction, trigger alerts instead of performing route optimization.

Alert Generation: Utilize the Accessibility subsystem to generate alerts tailored to the level of distraction detected. For example, for moderately distracted drivers (Class 2), the system may issue visual and auditory alerts advising the driver to pull over. For highly and severely distracted drivers (Class 3 and Class 4), the alerts may escalate in urgency, potentially including emergency notifications to authorities or emergency contacts.

Feedback Loop and Continuous Monitoring: Implement a feedback loop to continuously monitor the driver's distraction level and adjust the system's behaviour accordingly. Collect feedback from the driver's response to alerts, as well as additional sensor data if available, to refine the logic and improve the system's effectiveness over time.

Example Scenario:

Driver Distraction Level Detected: The Focus Monitor subsystem detects that the driver is minimally distracted (Class 1) due to briefly looking away from the road to adjust the radio.

Predicate Logic Evaluation: The distraction level (Class 1) meets the condition for using Dijkstra's Algorithm to find the optimal route.

System Response: The Route Adjustment subsystem invokes Dijkstra's Algorithm to calculate the optimal route and provides navigation instructions to the driver.

Route Optimization: The driver receives directions that minimize travel time and avoid traffic congestion, enhancing the overall driving experience while ensuring safety.

By using predicate logic to conditionally select the appropriate course of action based on the driver's distraction level, the system can prioritize safety by issuing alerts when necessary while still providing route optimization capabilities when the driver's distraction level is low.

For implementing FocusRide’s route optimization algorithm, we use Dijkstra’s algorithm in conjunction with the OpenStreet API. Dijkstra's algorithm is a classic pathfinding algorithm used to find the shortest path between nodes in a graph. In the context of route optimization for driving, each node represents a location or intersection, and edges represent roads connecting these locations. Here's how Dijkstra's Algorithm is applied:

1. **Graph Representation:** The GPS component provides information about the current location of the vehicle and nearby intersections or waypoints. This information forms the graph nodes. The Traffic Data component provides data about traffic density, which can be used to assign weights to the edges connecting nodes. Higher traffic density results in higher edge weights, representing slower travel times.

2. **Initialization:** The algorithm initializes a priority queue (such as a min-heap) to store nodes ordered by their tentative distances from the starting point (current location). Initially, only the starting node (current location) has a distance of 0; all other nodes have distances set to infinity.

3. **Exploration:** The algorithm iterates through the nodes in the priority queue, selecting the node with the smallest tentative distance. For each neighbouring node of the selected node, the algorithm calculates the tentative distance from the starting node through the selected node. If this tentative distance is smaller than the current distance stored for the neighbouring node, the distance is updated, and the neighbouring node is added to the priority queue.

4. **Termination:** The algorithm terminates when all nodes have been explored, or when the destination node is reached. At this point, the shortest path from the starting node to each node in the graph is known.

Simulation and Results

Overview: The simulation of the FocusRide system was designed to validate the effectiveness of the distracted driving behaviour monitoring and dynamic route optimization functionalities under various scenarios. Two key test scenarios were established: one with a non-distracted driver (Test Image 1) and another with a distracted driver (Test Image 2). These scenarios helped assess the system's response to different levels of driver focus.

Simulation Setup:

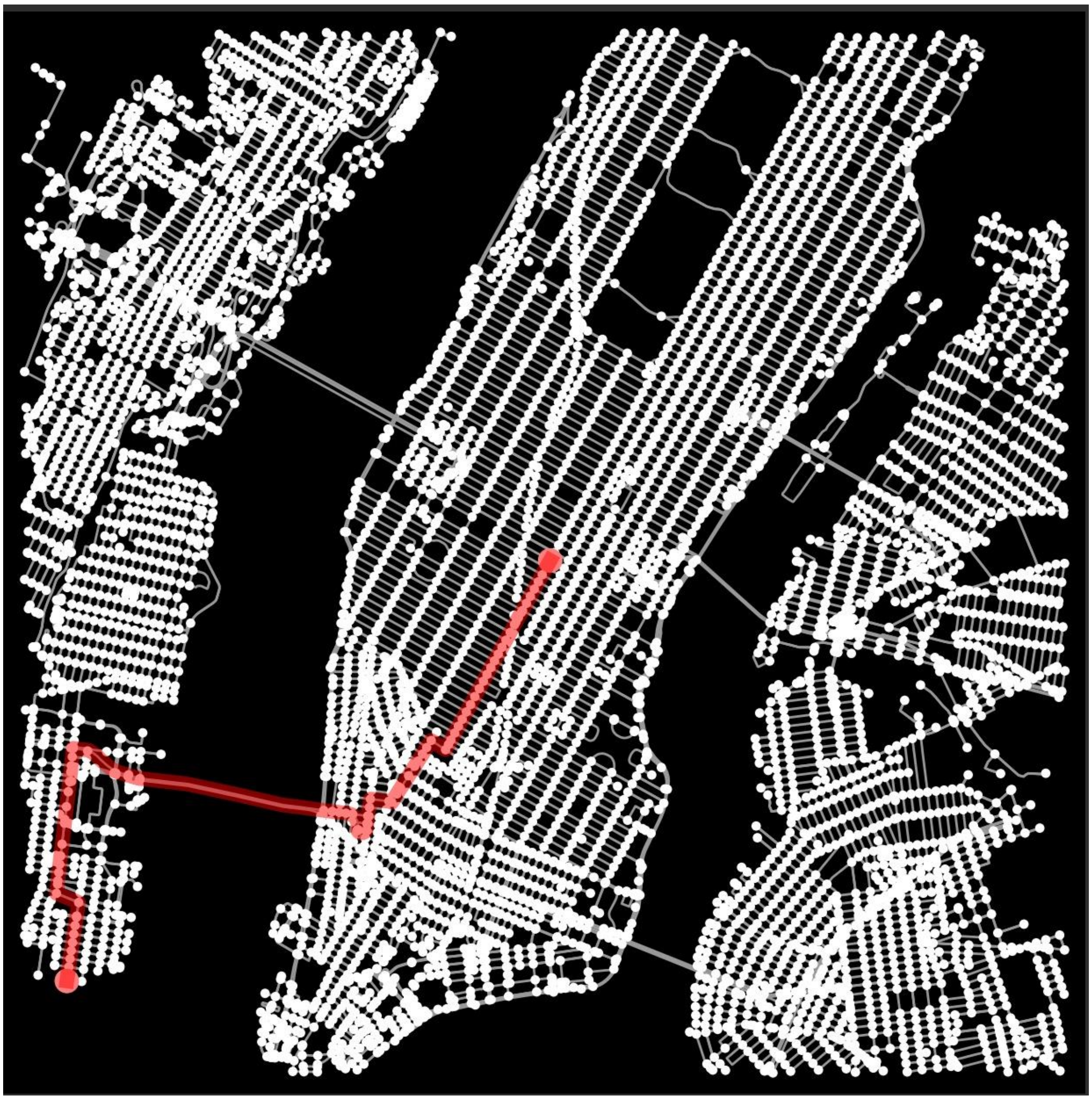
1. **Initialization:** The simulation initiates with the driver starting the vehicle and entering the destination either through physical input or voice command.
2. **Driver Focus Assessment:** Using high-resolution cameras and infrared sensors, the system captures the initial image of the driver. It assesses whether the driver is looking at the road with hands properly placed on the steering wheel.
3. **Route Optimization:** If no distraction is detected, the system proceeds to optimize the route using Dijkstra's algorithm, factoring in current traffic data and road conditions.
4. **Distraction Detection and Response:** During the journey, the system continuously monitors the driver. If distraction is detected (as determined by the deep learning model from the camera feed), the system engages predicate logic to evaluate the level of distraction and respond accordingly.

Test Scenarios and Results

Scenario 1: Non-Distracted Driving

Input: Test Image 1 showing the driver with focused attention.

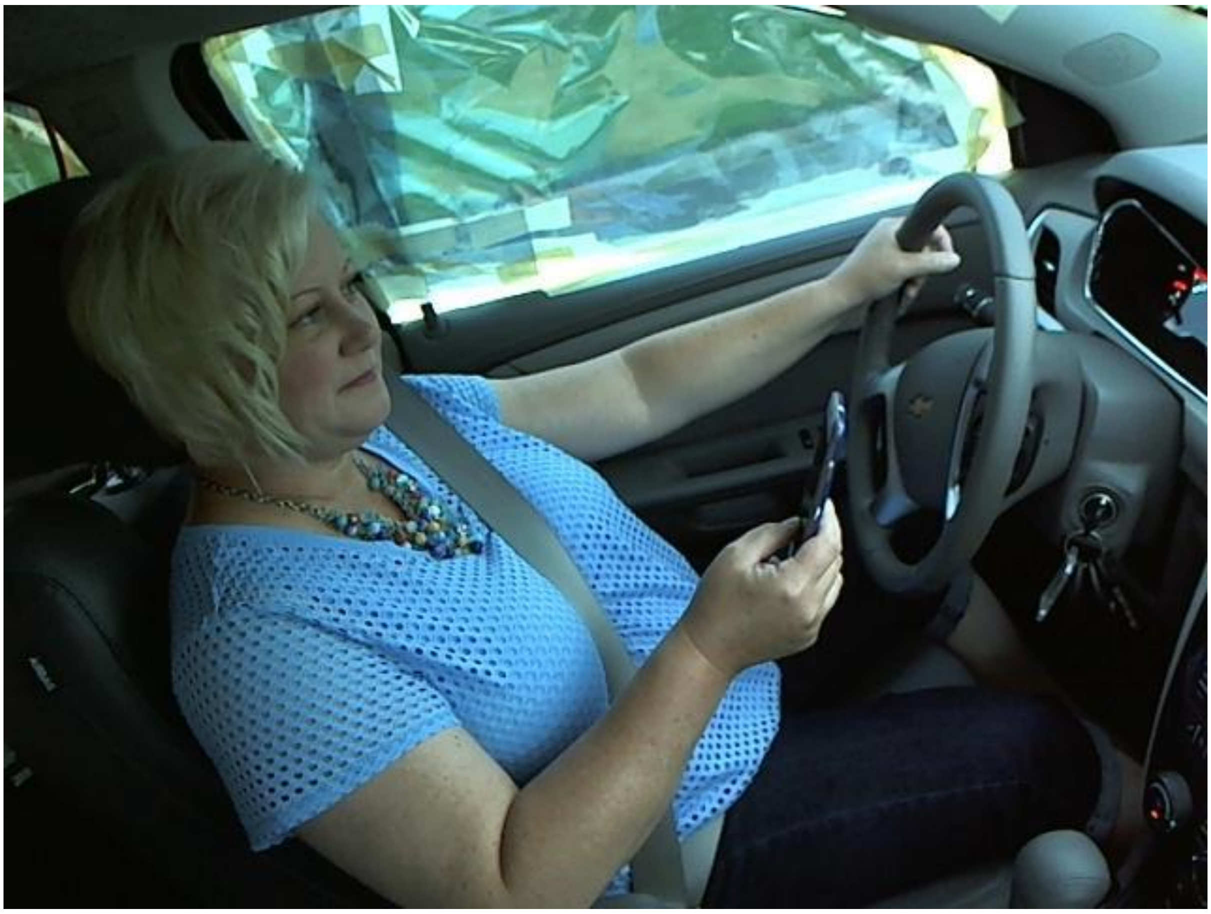
System Behavior: The FocusRide system detects minimal distraction. It optimizes the route, maintaining standard navigation. It looks something like this:



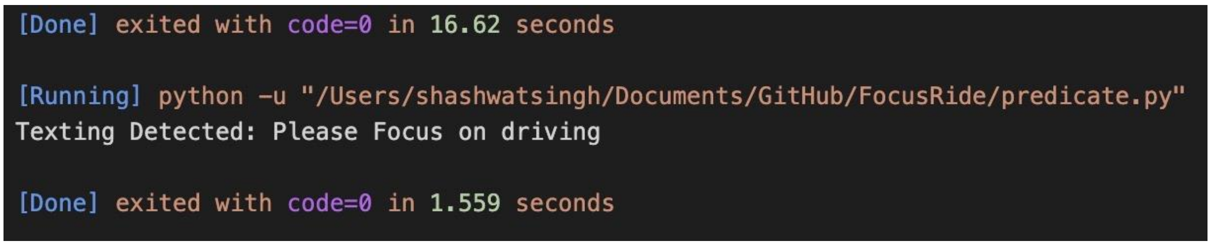
Outcome: The journey proceeds without alerts, demonstrating the system's capability to accurately recognize a focused driver state and optimize the route efficiently.

Scenario 2: Distracted Driving

Input: Test Image 2 showing the driver distracted (e.g., looking away from the road, handling a phone).



System Behaviour: The system identifies significant distraction. Based on the distraction level (Class 3), it immediately issues auditory and visual alerts urging the driver to regain focus or pull over.



```
[Done] exited with code=0 in 16.62 seconds

[Running] python -u "/Users/shashwatsingh/Documents/GitHub/FocusRide/predicate.py"
Texting Detected: Please Focus on driving

[Done] exited with code=0 in 1.559 seconds
```

Outcome: The alerts effectively prompt the driver to adjust their behaviour, highlighting the system's potential to enhance road safety by mitigating risky driving practices.

Effectiveness of the System: The results from these simulations validate that FocusRide can enhance both safety and route efficiency by:

1. Accurately detecting levels of driver distraction and providing timely interventions.

2. Optimizing travel routes that adapt to the driver's current state, thereby reducing the risk of accidents due to distracted driving.
3. This rigorous testing framework not only proves the operational effectiveness of FocusRide but also underscores its potential as a transformative technology in smart transportation solutions.

Conclusion

Contributions to Smart Transportation Solutions:

The "FocusRide" project represents a significant advancement in the realm of smart transportation systems, addressing crucial aspects of road safety and efficiency through the integration of artificial intelligence and real-time data processing. This system sets a new standard by not only monitoring driver behaviour to detect signs of distraction but also dynamically adjusting routes to accommodate the driver's current state, thus significantly reducing the risk of accidents caused by inattention.

FocusRide's implementation of sophisticated algorithms and multi-modal feedback mechanisms has demonstrated the practical viability of using advanced technologies to enhance driver safety. The project has successfully developed a prototype that provides real-time, context-aware navigation aids, which is a pioneering step towards more responsive and adaptive transportation systems.

Impact on Future Transportation Technologies:

FocusRide's approach to integrating driver focus monitoring with route optimization has potential implications beyond individual vehicle safety. As the system matures, it could serve as a foundational technology for autonomous vehicle systems, where understanding driver state is crucial for the handoff between automated and manual control. Moreover, the technologies developed through FocusRide could be adapted for broader applications in traffic management and urban planning, contributing to smarter, safer urban environments.

The project also opens up possibilities for enhancing vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications, enabling cars to share information about driver state and road conditions in real-time, thereby enhancing collective safety on the road.

Potential Areas for Further Development and Research:

While FocusRide has laid a robust foundation for safety and efficiency in transportation, several areas warrant further exploration:

Integration with Wider Transportation Ecosystems: Future research could explore how FocusRide can be integrated into broader smart city infrastructures, including public transportation systems and traffic management networks.

Advanced Prediction Algorithms: Developing more sophisticated algorithms to predict potential distraction behaviours before they occur could further enhance the system's preventative capabilities.

User Experience Optimization: Continuous research into user interaction with the FocusRide system can help refine its usability and accessibility, ensuring that it meets the diverse needs of all drivers, particularly those with specific disabilities.

Ethical and Privacy Concerns: As FocusRide involves monitoring driver behaviour, ongoing research into data privacy, and ethical considerations is crucial to address potential concerns and develop frameworks for user data protection.

Scalability and Customization: Investigating the scalability of the system to different vehicle types and driving environments, as well as allowing more customized settings for individual user preferences and needs, could broaden its applicability and user acceptance.

References

1. Distracted driving statistics from CDC: https://www.cdc.gov/transportationsafety/distracted_driving/index.html
2. Distracted driving in 2022: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813559
3. Kaggle: Driver Behaviour dataset: https://www.kaggle.com/datasets/robinreni/revitsone-5class
4. Texting and Driving Statistics 2024: https://www.forbes.com/advisor/car-insurance/texting-driving-statistics/
5. Driver behaviour detection and classification using deep convolutional neural networks: https://www.sciencedirect.com/science/article/pii/S095741742030066X
6. A hybrid neural network for driving behaviour risk prediction based on distracted driving behaviour data: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0263030#:~:text=The%20Driving%20Behavior%20Risk%20Prediction,the%20distracted%20driving%20behavior%20data.
7. Deep learning-based image recognition for autonomous driving: https://www.sciencedirect.com/science/article/pii/S0386111219301566