Smart City Road Safety Monitoring System

Smart City AI Industry Project

COSC3P71: Introduction to Artificial Intelligence
Instructor: Dr. Beatrice Ombuki-Berman
Project TAs: Rasa Khosrowshahli and Zachary McGovarin

1 Project Overview

This project introduces computer science students to practical applications of artificial intelligence in smart city infrastructure. Students will develop an intelligent road safety monitoring system that combines computer vision, machine learning, and graph algorithms to create a comprehensive solution for urban transportation safety.

The project simulates real-world challenges faced in smart city development, where AI systems must process visual data, make intelligent decisions, and provide actionable insights for city planning and navigation systems.

2 Learning Objectives

Upon completion of this project, students will:

- Gain hands-on experience with computer vision and deep learning frameworks
- Understand the practical applications of pre-trained neural networks
- Learn to work with geospatial data and mapping technologies
- Implement graph algorithms for pathfinding and route optimization
- Develop skills in data collection, annotation, and model fine-tuning
- Experience the full pipeline of an AI project from data to deployment

3 Project Components

3.1 Component 1: Data Collection and Annotation

Objective: Collect and prepare a comprehensive dataset of road conditions from a chosen area.

Tasks:

1. **Area Selection:** Choose a small neighborhood or district (approximately 1-2 square kilometers) for data collection
2. **Image Capture:** Using available equipment (dashcam, smartphone, handheld camera, or drone), systematically capture images of:
   - Road surfaces and conditions
   - Traffic signs and signals
   - Intersections and crosswalks

   - Potential hazards (potholes, construction zones, debris)
   - Different lighting and weather conditions

3. **Data Annotation:** Label each image with:
   - GPS coordinates (latitude, longitude)
   - Timestamp
   - Road condition classification (safe, minor issues, major problems)
   - Specific problem types (if applicable): potholes, cracks, flooding, construction, debris
   - Direction of travel

4. **Dataset Organization:** Structure data in a format suitable for machine learning training

Deliverables:

- Minimum 500 annotated images
- CSV file with metadata and labels
- Documentation of data collection methodology

3.2 Component 2: Geospatial Mapping and Graph Construction

Objective: Create a digital map representation of the surveyed area using collected images and coordinate data.

Tasks:

1. **Image Mapping:** Use mapping software (such as QGIS, Google Earth Pro, or custom Python scripts with libraries like Folium) to:
   - Plot image locations on a map using GPS coordinates
   - Visualize the coverage area and identify gaps
   - Create a visual overlay of road conditions

2. **Graph Construction:** Build a graph representation where:
   - Nodes represent intersections or significant points
   - Edges represent road segments between nodes
   - Edge weights incorporate distance and safety scores
   - Include bidirectional edges for two-way streets

3. **Coordinate System Integration:** Ensure proper coordinate system alignment for accurate distance calculations

Deliverables:

- Interactive map visualization
- Graph data structure (adjacency list or matrix)
- Documentation of mapping methodology

3.3 Component 3: AI Model Development and Fine-tuning

Objective: Implement and customize a computer vision model for road condition assessment.

Tasks:

1. **Model Selection:** Use a pre-trained convolutional neural network (recommended: YOLOv5, YOLOv8, or similar object detection model)
2. **Data Preparation:**
   - Split dataset into training (70%), validation (15%), and testing (15%) sets
   - Implement data augmentation techniques (rotation, brightness adjustment, cropping)
   - Format data according to model requirements
3. **Model Fine-tuning:**
   - Configure the model for binary classification (safe vs. hazardous roads)
   - Implement multi-class classification for problem types
   - Train the model using transfer learning techniques
   - Monitor training metrics and prevent overfitting
4. **Model Evaluation:**
   - Calculate accuracy, precision, recall, and F1-score
   - Generate confusion matrices
   - Test on unseen data from the test set

Deliverables:

- Trained model files
- Training and evaluation scripts
- Performance metrics and analysis report
- Model inference pipeline

3.4 Component 4: Intelligent Navigation System

Objective: Develop a navigation system that finds safe routes and adapts to real-time road conditions.

Tasks:

1. **Safe Path Algorithm:** Implement pathfinding algorithms:
   - Dijkstra's algorithm for shortest safe path
   - A* algorithm for optimized pathfinding
   - Custom weight function combining distance and safety scores
2. **Route Planning Interface:**
   - Create a user interface for selecting start and end points
   - Display multiple route options with safety ratings
   - Show estimated travel time and distance

3. **Real-time Adaptation:**
   - Simulate real-time problem detection using the trained model
   - Implement dynamic re-routing when new problems are detected
   - Update graph weights based on new information
   - Provide alternative route suggestions to users
4. **User Notification System:**
   - Alert users about detected road problems
   - Suggest alternative routes with explanations
   - Provide estimated time savings or delays

Deliverables:

- Navigation system implementation
- User interface (web-based or desktop application)
- Real-time detection and re-routing demonstration
- System architecture documentation

4 Technical Requirements

4.1 Software and Tools

- **Programming Language:** Python 3.8+
- **Deep Learning Framework:** PyTorch or TensorFlow
- **Computer Vision:** OpenCV, PIL/Pillow
- **Machine Learning:** scikit-learn, NumPy, pandas
- **Mapping and Visualization:** Folium, matplotlib, QGIS (optional)
- **Graph Algorithms:** NetworkX
- **Web Framework:** Flask or Streamlit (for user interface)
- **Data Storage:** SQLite or PostgreSQL

4.2 Hardware Requirements

- Computer with GPU support (recommended for model training)
- Camera device (smartphone, dashcam, or drone)
- GPS-enabled device for coordinate collection
- Minimum 8GB RAM, 50GB storage space

5 Assessment Criteria

5.1 Technical Implementation

- Quality and completeness of dataset
- Model performance and accuracy
- Correctness of graph algorithms
- System integration and functionality

5.2 Documentation and Presentation

- Code documentation and comments
- Technical report quality
- Final presentation clarity
- Methodology explanation

5.3 Innovation and Problem-Solving

- Creative solutions to challenges
- Additional features or improvements
- Handling of edge cases
- System robustness

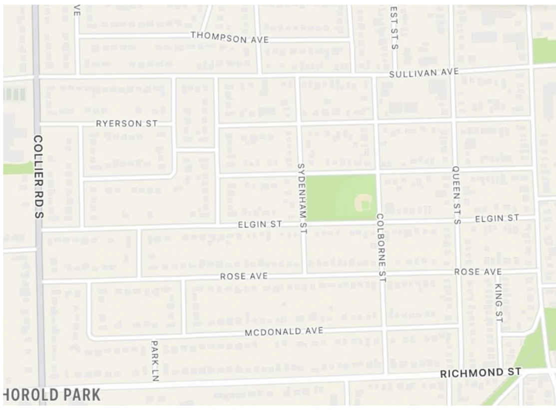
6 Expected Outcomes

Students will develop a working prototype of an intelligent road safety monitoring system that demonstrates:

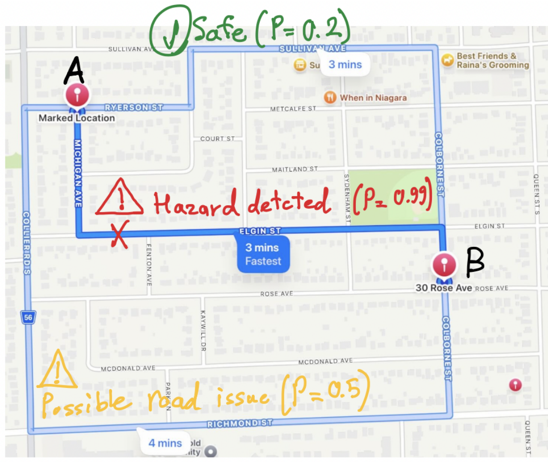
- Automated detection of road problems using computer vision
- Intelligent route planning based on safety considerations
- Real-time adaptation to changing road conditions
- Integration of multiple AI technologies in a practical application

6.1 Visual Demonstration

The expected functionality of the smart city road safety monitoring system is demonstrated through side-by-side visualizations and a safety legend:



(a) Baseline street network



(b) Routes with safety classifications

🟢	Safe	$p < 0.3$
🟠	Possibly Hazardous	$0.3 \leq p < 0.7$
🔴	Hazardous	$p \geq 0.7$

(c) Safety legend for route classification

Figure 1: (a) Raw map of the selected urban area. (b) Intelligent route planning output with safety-based classifications and travel time indicators. A legend summarizes probability thresholds used for safety categories.

As shown in Figure 1a, the system begins with a comprehensive map of the urban area, capturing all road segments, intersections, and infrastructure elements. This baseline data serves as the foundation for the intelligent navigation system.

Figure 1b demonstrates the system's intelligent navigation capabilities through the route classification system, where each potential path is evaluated using probability-based safety scoring. The visualization shows three distinct routes from point A to point B, each with different safety classifications based on the probability values: safe routes ($p < 0.3$), possibly hazardous routes ($0.3 \leq p < 0.7$), and hazardous routes ($p \geq 0.7$). This approach allows users to make informed decisions about their travel routes based on real-time road condition assessments, with visual indicators clearly showing travel times and potential road issues.

This project provides students with valuable experience in applying AI techniques to solve real-world urban challenges, preparing them for careers in smart city development, autonomous systems, and AI engineering.

7 Additional Resources

- YOLOv5 Documentation: `https://github.com/ultralytics/yolov5`
- OpenCV Tutorials: `https://opencv.org/`
- NetworkX Documentation: `https://networkx.org/`
- Folium Mapping: `https://python-visualization.github.io/folium/`
- QGIS User Guide: `https://qgis.org/`