Simulation Of Autonomous Systems Workshop
Final Project Design Document

Roei Levi

August 4, 2025

1 Motivation

In urban environments, autonomous vehicles must navigate complex interactions with pedestrians, where behavior is often unpredictable. Ensuring safety in these scenarios is essential for preventing accidents and gaining public trust.

This project, implemented in the CARLA simulation environment, aims to develop a system that enables autonomous cars to detect and respond to pedestrian movement in real time, with a focus on preventing collisions. Improving pedestrian safety is a crucial step toward the reliable and responsible deployment of self-driving vehicles.

2 Project Requirements

2.1 Functional Requirements

1. Detect pedestrians using sensor data within the simulation.
2. Track detected pedestrians across frames to maintain identity over time.
3. Estimate the motion of each pedestrian (e.g., position, velocity).
4. Predict potential collisions between pedestrians and the ego vehicle.
5. Trigger appropriate vehicle responses (e.g., braking) to avoid collisions.

2.2 Technical Requirements

1. Integrate an object detection model (e.g., YOLO) for identifying pedestrians in camera frames.
2. Use a multi-object tracking algorithm (e.g., DeepSORT) to track pedestrians over time.
3. Implement a motion estimation model (e.g., Kalman Filter) to estimate pedestrian trajectories using sensor data (e.g. LiDAR).

4. Develop a collision prediction module that evaluates future intersections of paths.
5. Design an agent to control the ego vehicle's response based on predicted collisions.
6. Use CARLA as the simulation environment for development and testing.

2.3 Constraints

1. The simulation is limited to the capabilities of the CARLA simulator.
2. Real-time performance and model complexity are constrained by available hardware resources.
3. The entire system must be completed within the project's deadline.
4. The project scope is limited to interactions with pedestrians; other road users are excluded.
5. The project scope assumes that all sensors function correctly and do not fail while the car is driving.

3 System Architecture and Key Components

This section outlines the structure of the system, describing the key components and their interactions. The architecture is designed to detect pedestrians, track their motion, estimate future positions, and determine potential collisions within the CARLA simulation environment. The system is composed of modular components that interact to form a complete perception and decision-making pipeline for autonomous navigation.

1. **Camera and LiDAR**: The primary sensors used to perceive the environment. The camera captures RGB images while the LiDAR provides 3D point cloud data.
2. **Sensor Manager**: Responsible for managing and synchronizing all sensors in the CARLA simulation. It is the interface that downstream modules use to get the sensor data they need.
3. **Object Detector**: Applies a deep learning model (e.g., YOLO) to camera images in order to identify and localize pedestrians via bounding boxes.
4. **Object Tracker**: Associates detected objects across consecutive frames to maintain persistent identities (e.g., using DeepSORT), enabling multi-object tracking.
5. **Motion Estimator**: Computes the velocity and position of each tracked object over time using state estimation techniques such as a Kalman filter.

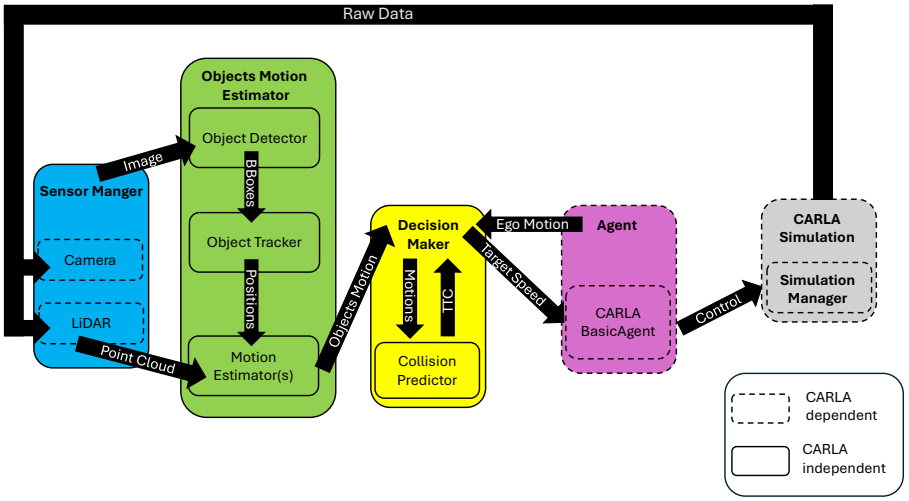

Figure 1: High-level system architecture diagram of the project.

6. **Objects Motion Estimator**: A composite module that integrates the sensor data with the object detector, object tracker, and motion estimator to maintain an up-to-date state of all nearby pedestrians.

7. **Collision Predictor**: Uses the outputs of the Objects Motion Estimator along with the ego vehicle's trajectory to identify potential future collisions.

8. **Decision Maker**: Consumes predictions from the Collision Predictor and outputs high-level commands, including the target speed and whether an emergency stop is required.

9. **CARLA Basic Agent**: A navigation agent provided by the CARLA library that follows waypoints and traffic rules to drive the vehicle (*but doesn't avoid pedestrians*). This component itself is not modified or in this project.

10. **Agent**: A wrapper around the Basic Agent that integrates with the rest of the system. It uses the outputs of the Decision Maker (target speed and emergency stop signals) to override the Basic Agent's behavior, enabling the vehicle to avoid pedestrians. *Note: the Collision Aware Agent is one of the two modules directly dependent on CARLA.*

11. **Simulation Manager**: Responsible for running and managing the CARLA simulation itself, including the ego vehicle and pedestrians. *Note: this is the second module directly dependent on CARLA.*

Design Justification

The system has been designed according to the SOLID principles of object-oriented design, ensuring maintainability, extensibility, and modularity:

1. **Single Responsibility Principle (SRP)**: Each module in the system is responsible for a single, well-defined task. For instance, the *Sensor Manager* handles sensor data requests, while the *Object Detector*, *Object Tracker*, and *Motion Estimator* each encapsulate their respective functionalities.

2. **Open–Closed Principle (OCP)**: The architecture is modular and open to extension. Components such as the object detection and tracking models can be replaced or upgraded without modifying the core system logic, provided they adhere to the predefined interfaces.

3. **Liskov Substitution Principle (LSP)**: Each component is implemented against abstract interfaces, allowing interchangeable use of alternative implementations. For example, the object detection module may use different models (e.g., YOLO or Faster R-CNN) as long as they conform to the detection interface.

4. **Interface Segregation Principle (ISP)**: Interfaces are designed to be minimal and role-specific, ensuring that modules depend only on the methods they actually use. This promotes loosely coupled components and reduces unnecessary dependencies.

5. **Dependency Inversion Principle (DIP)**: The system explicitly depends on abstractions rather than concrete classes. Core components such as the *Agent* and *Objects Motion Estimator* interact with their dependencies via abstract interfaces, promoting testability and flexibility in implementation choices.

Overall, these design decisions ensure that the system is robust to change, easy to test, and reusable in different autonomous simulation contexts. By adhering to abstraction-based development, the system can evolve without introducing regressions in unrelated modules.

4 Description of Simulation Scenarios

The following scenarios were originally planned to evaluate the system's performance in common pedestrian-related situations:

1. **Straight Crosswalk Scenario** – A pedestrian crosses directly in front of the ego vehicle at a marked crosswalk.

2. **Sudden Pedestrian Entry from Sidewalk** – A pedestrian abruptly steps into the road from the sidewalk.

3. **Multiple Sequential Pedestrians** – Several pedestrians cross one after another, requiring sustained tracking and response.

4. **Static Pedestrian on Road** – A pedestrian remains stationary in the vehicle's path.

5. **Pedestrian Walking on Sidewalk (No Crossing)** – Ensures the vehicle doesn't stop when pedestrians are not entering the road.

6. **Pedestrian Standing at Edge of Crosswalk** – Tests system's decision-making when a pedestrian is near the crosswalk but not necessarily crossing.

However, creating controlled setups proved time-intensive and not representative of the variability in real environments. Instead, a more holistic approach was adopted: a large number of pedestrians were spawned with random destinations, and the ego vehicle was given its own destination. This setup implicitly exercised scenarios 1, 3, 5, and 6 while also providing a richer and more realistic evaluation environment, much closer to real-world production conditions, than individually staged scenarios.

5 Performance Metrics

To evaluate the effectiveness and efficiency of the proposed system, the following key performance indicators (KPIs) are measured within the CARLA simulation environment:

1. **Collision Count** – The total number of collisions between the ego vehicle and pedestrians throughout the simulation.

2. **Unsafe Interactions** – The number of pedestrian interactions in which the distance between the ego vehicle and a pedestrian falls below or equal to a safety threshold (e.g., 1 meter) at any point. A lower value indicates safer behavior.

3. **Minimum Time-To-Collision (TTC)** – The lowest *system-estimated* TTC observed across the entire simulation.

4. **Average TTC on Predicted Collisions** – For all situations in which the system predicted a collision, the average *system-estimated* TTC at the time of prediction is reported. Higher values indicate earlier predictions and more reaction time.

5. **System Latency** – The time taken between receiving a new sensor frame and issuing a control decision. While this is hardware-dependent, it is measured and reported to identify potential performance issues.

6 Challenges and Risks

Several technical and conceptual challenges may arise during the development of the system. Below are the most critical ones along with proposed mitigation strategies:

1. **Sudden Pedestrian Entry at Close Range**
   Pedestrians entering the roadway abruptly and at short distances can be extremely difficult to handle in time. To mitigate this, the system will include an emergency stop mechanism triggered when the estimated time-to-collision (TTC) drops below a critical threshold.

2. **High Velocity Estimates from Reappearing Objects**
   When objects go out of frame and are re-identified after a delay, motion estimators may compute unrealistically high velocities. The solution is to reset an object's motion state if it is not observed for a fixed duration.

3. **Sensor Synchronization**
   Inconsistent timestamps between the camera and LiDAR sensors can cause inaccurate fusion results. This is handled by making sure all sensors start listening at the same time and the simulation is ticked only after each sensor has collected its data.

7 Evaluation Criteria

The system's success will be assessed using the defined performance metrics in a qualitative and relative manner. Some metrics, such as system latency and TTC estimates, are hardware and context dependent, so strict numeric thresholds will not be enforced for them.

- **Collision Count**: Should be zero across the entire simulation. Any collision indicates a critical failure of the system.

- **Unsafe Interactions**: Should be zero across the entire simulation. Any unsafe interaction (distance $\leq$ safety threshold) indicates the vehicle came unacceptably close to a pedestrian and therefore a failure in maintaining safe clearance.

- **Minimum TTC**: The lowest *system-estimated* Time-To-Collision values should reflect timely responses. TTC values as high as possible are strived for, but this is an academic project with limited time and scope, so perfect optimization is not expected. Furthermore, the TTC values are estimated by the system and may not exactly reflect the ground truth due to sensor noise, estimation errors, or simplifying assumptions. For these reasons, no strict numeric threshold is enforced; instead, the values are interpreted in context—consistently very low TTCs would indicate delayed reactions.

- **Average TTC on Predicted Collisions**: Higher average *system-estimated* TTCs in cases where collisions were predicted indicate earlier predictions. As with the minimum TTC, these are system estimates rather than ground truth measurements, and while higher values are preferable, no strict cut-off is imposed.
- **System Latency**: Since latency depends heavily on the hardware used, no strict threshold will be set. Instead, latency will be measured and reported for awareness. Excessive delays that noticeably impact decision-making or cause frame skipping may be flagged as performance issues.

8 Timeline

- **Weeks 1–2**: Finalize system architecture. Set up Sensor Manager and validate camera and LiDAR data collection in CARLA.
- **Weeks 3–4**: Integrate object detection (e.g., YOLO) and multi-object tracking (e.g., DeepSORT) to enable consistent pedestrian tracking.
- **Weeks 5–6**: Implement and test motion estimation (e.g., Kalman Filter). Combine detection, tracking, and estimation into a unified motion estimation module.
- **Weeks 7–8**: Develop collision prediction and reactive vehicle control agent based on predicted pedestrian paths.
- **Weeks 9–10**: Design and test simulation scenarios. Evaluate system performance and address integration issues.
- **Weeks 11–12**: Finalize system and ensure the evaluation criteria is met.
- **Week 13**: Deliver final presentation and submit project.