NATIONAL CENTER FOR UNDERSTANDING FUTURE
TRAVEL BEHAVIOR AND DEMAND

Final Project Report

Vehicle Edge Computing for Travel Behavior and Demand in Future Intelligent Transportation Systems

BY

Rafael Trinidad
Email: retrinidad@cpp.edu

Michael Ly
Email: michaelly@cpp.edu

Yichi Cheng
Email: yichicheng@cpp.edu

Yunsheng Wang, PhD
Email: yunshengwang@cpp.edu

Yongping Zhang, PhD
Email: yongpingz@cpp.edu

Wen Cheng, PhD
Email: wcheng@cpp.edu

California State Polytechnic University, Pomona
3801 W Temple Ave, Pomona, CA 91768

January 2026

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No. N/A	**2. Government Accession No.** N/A	**3. Recipient's Catalog No.** N/A	
4. Title and Subtitle Vehicle Edge Computing for Travel Behavior and Demand in Future Intelligent Transportation Systems (ITS)		**5. Report Date** January 2026 **6. Performing Organization Code** N/A	
7. Author(s) Rafael Trinidad, https://orcid.org/0009-0003-6234-1310 Michael Ly Yichi Cheng Yunsheng Wang, PhD, https://orcid.org/0000-0002-9876-5356 Yongping Zhang, PhD, https://orcid.org/0000-0002-5935-3834 Wen Cheng, PhD, https://orcid.org/0000-0001-7225-6169		**8. Performing Organization Report No.** N/A	
9. Performing Organization Name and Address California State Polytechnic University, Pomona 3801 W Temple Ave, Pomona, CA 91768		**10. Work Unit No. (TRAIS)** N/A **11. Contract or Grant No.** 69A3552344815 and 69A3552348320	
12. Sponsoring Agency Name and Address U.S. Department of Transportation, University Transportation Centers Program, 1200 New Jersey Ave, SE, Washington, DC 20590		**13. Type of Report and Period Covered** Final Report, 2024-2025 **14. Sponsoring Agency Code** USDOT OST-R	
15. Supplementary Notes Conducted in cooperation with the U.S. Department of Transportation, Federal Highway Administration.			

16. Abstract

The future of transportation faces critical challenges due to computational constraints in autonomous vehicles and limitations of cloud-based solutions. This project introduces a vehicular edge computing platform that addresses these challenges by leveraging small-scale autonomous vehicles (Donkey Cars) and edge computing nodes. We implement a dual-protocol communication system where image data streams via UDP while control commands are transmitted through TCP, ensuring both speed and reliability. Our platform demonstrates significant performance improvements in real-time processing and decision-making capabilities, achieving up to 99.75% reduction in inference latency compared to local processing, while reducing CPU usage by 66.5% and memory utilization by 68.5%.

A second two-vehicle multi-threaded experiment also demonstrates great performance with CPU usage remaining around 20%, consistent memory usage at 0.8%, and inference latency around 10 milliseconds. The results of both experiments suggest that edge computing integration could be a viable solution for future intelligent transportation systems, particularly in scenarios requiring real-time hazard detection and traffic management. Our results reveal significant reductions in inference latency, with future suggested research focusing on GPU acceleration, CHI@EDGE, and more.

17. Key Words Edge Computing; Autonomous Vehicles; Real-time Processing; Latency Reduction		**18. Distribution Statement** No restrictions.	
19. Security Classif.(of this report) Unclassified	**20. Security Classif.(of this page)** Unclassified	**21. No. of Pages** 23	**22. Price** N/A

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, under Grant No. 69A3552344815 from the U.S. Department of Transportation's University Transportation Centers Program. The U.S. Government assumes no liability for the contents or use thereof.

ACKNOWLEDGMENTS

This research was supported by the National Center for Understanding Future Travel Behavior and Demand (TBD), a National University Transportation Center sponsored by the U.S. Department of Transportation (USDOT) under grant number 69A3552344815.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

EXECUTIVE SUMMARY

The rapid advancement of autonomous vehicles promises increased safety, efficiency, and accessibility in transportation. However, the high computational demands of onboard processing units and the latency limitations of cloud-based solutions present significant challenges for large-scale deployment. This project introduces a low-cost, open-source vehicular edge computing platform designed to address these constraints by combining small-scale autonomous vehicles (Donkey Cars) with edge computing nodes.

The platform employs a dual-protocol communication system: image data streams from vehicles via UDP for low-latency transmission, while control commands are sent via TCP to ensure reliability. Experiments demonstrate substantial performance improvements compared to local processing on the Raspberry Pi 4. For single-vehicle trials, inference latency decreased by up to 99.75%, CPU usage dropped by 66.5%, and memory utilization fell by 68.5%. Multi-vehicle, multi-threaded tests further confirmed the platform's scalability, maintaining CPU usage around 20%, memory usage at 0.8%, and inference latency near 10 milliseconds.

The platform integrates low-cost hardware, including Raspberry Pi 4-equipped Donkey Cars and NVIDIA Jetson AGX Orin edge servers, leveraging TensorFlow/Keras models for autonomous navigation. Its open-source architecture enables flexible experimentation and development, providing a foundation for broader adoption of autonomous driving technologies without relying on expensive sensors or computing units.

This research highlights vehicle edge computing as a viable solution for real-time processing, hazard detection, and traffic management in intelligent transportation systems. Future directions include incorporating GPU acceleration, expanding to larger multi-vehicle setups, evaluating C-V2X communications and more advanced perception models, and exploring cloud-managed testbeds such as CHI@EDGE. By bridging affordability with computational efficiency, this platform paves the way for scalable, accessible, and high performance autonomous vehicle research and deployment.

INTRODUCTION

Self-driving autopilot technology has the potential to make transportation safer, more accessible, and efficient. It can reduce human errors, the leading cause of accidents, thereby saving lives and preventing serious injuries. It also offers mobility to individuals unable to drive, optimizes traffic flow, and reduces congestion costs, resulting in significant savings in time and fuel. These fuel savings further promote environmental sustainability through energy-efficient driving and integration with electric vehicles (Anderson et al., 2016).

The rapid advancement of autonomous driving technologies, particularly in the realm of Advanced Driver-Assistance Systems (ADAS), has brought substantial improvements in vehicle safety and driving automation. However, the majority of current autonomous vehicle systems rely on expensive, onboard computing units as the primary processing platform for sensor data. These on-board units, though powerful, face significant limitations, including high energy consumption, limited storage capacity, and sensitivity to extreme temperatures, making them less suitable for large-scale deployment and adaptation to different vehicle types (e.g., traditional vehicles with minimal sensor suites). Moreover, these onboard systems are typically paired with costly and complex sensors such as LiDAR, cameras, and radar, which not only elevate vehicle production costs but also create challenges for retrofitting older vehicle models (Lu et al., 2023; Mallozzi et al., 2019; Bojarski et al., 2016).

Another prevailing approach to handling the intensive computational demands of autonomous driving is cloud computing. By leveraging the vast computational resources of centralized data centers, cloud computing provides a scalable solution for data storage and processing (Olariu, 2020). However, the cloud-based paradigm is severely hampered by substantial drawbacks, primarily the high latency in communication between vehicles and remote servers, which introduces delays that are critical for real-time decision making. Furthermore, significant bandwidth requirements for transmitting large amounts of sensor data impose constraints that can affect the efficiency and safety of autonomous operations (Liu et al., 2019).

Given these challenges, there is a pressing need for alternative solutions that combine the benefits of local processing with low-cost deployment. One promising approach is to leverage vehicle edge computing (Shi et al., 2016), where processing power is moved closer to the vehicle's environment, specifically, through the installation of edge nodes at traffic lights and other urban infrastructure points. This decentralized edge computing platform allows for real-time data processing, reduces the reliance on expensive on-board computing hardware, and minimizes communication latency by facilitating local decision-making. Additionally, this approach can enable the integration of ADAS technologies into vehicles that may not be equipped with high-end sensors or onboard computing systems, thus promoting the adoption of autonomous features in a wider range of vehicles.

While vehicle edge computing is increasingly being researched, there is still more work to be done in making testbed resources and the development of such technology more openly available. Pre-existing resources and research are difficult to access as they are primarily proprietary tools kept from public view by major autonomous vehicle companies. However, there is a clear need for such a resource, as by 2025, nearly 400 million vehicles worldwide are projected to have connectivity features while 42% of consumer companies have cited cost as a barrier to adopting cloud solutions (Automotive Edge Computing Consortium, 2021; Deloitte, 2023). Our goal was to develop an affordable, open-source edge computing platform.

This project focused on creating a low-cost, open-source vehicle edge computing platform designed to address the limitations of current autonomous vehicle systems. By utilizing roadside

infrastructure and distributed computing resources, we present a scalable solution that enhances the capabilities of ADAS, increases accessibility, and reduces the overall cost of autonomous vehicle technology. Through this approach, we aim to not only improve the performance of autonomous systems but also democratize access to these innovations across a broader spectrum of vehicles and communities. The major contributions of our work are as follows:

- We design and implement a low-cost open-source vehicle edge computing platform.
- We conduct several experiments on our proposed vehicle edge computing platform.

LITERATURE REVIEW

Several research efforts have explored task offloading strategies and efficient computation techniques that can inform the development of low-cost vehicle edge computing platforms. This chapter presents and discusses representative studies in this area.

Fan et al. proposed a joint task offloading and resource allocation scheme to minimize the total task processing delay of all the vehicles via optimal task scheduling, channel allocation, and computing resource allocation for the vehicles and roadside units. The Generalized Benders Decomposition and Reformulation Linearization methods are used to solve the optimization problem. Vehicles can also flexibly choose to process tasks locally or offload their tasks to a roadside unit via the vehicle-to-infrastructure (V2I) mode or other vehicles via vehicle-to-vehicle (V2V) mode, using task computing services whenever needed, demonstrating adaptability for resource-constrained onboard units (Fan et al., 2023).

Feng et al. introduced a reverse offloading framework that can fully utilize the vehicular computational resources to relieve the burden of strained vehicular edge computing servers and further reduce the system latency. This framework utilizes binary reverse offloading and partial reverse offloading strategies for non-partitioned and partitioned tasks. These strategies use low-complexity Greedy-Based Efficient Searching (GES) algorithm and the Joint Alternative Optimization-Based Bi-Section Searching (JAOBSS) algorithm for task scheduling. Simulation results display that the GES algorithm is able to simultaneously attain optimal performance and low-complexity, offering promising applications for resource-limited environments, such as the Donkey Car (Fang et al., 2022).

Yin et al. emphasized the importance of hybrid offloading models, where vehicles delegate tasks to roadside units (RSUs) or share resources with other vehicles. An adaptive type selection algorithm based on the multi-armed bandit theory is used to maximize benefits derived from task vehicles, RSUs, and shared resource vehicles. This research underscores the value of open-source implementations as a means to foster collaboration and innovation. Building on this concept, our work adopts an open-source approach to further advance research and development in vehicle edge computing (Yin et al., 2024).

Zhang et al. proposed a distributed deep reinforcement learning-based quantization level allocation scheme for reducing latency in Federated Learning-enabled vehicle edge computing. For vehicular artificial intelligence applications, the gradients of vehicles' local models tend to be large, leading to large per-round latency when shared. Since gradient sharing serves as an alternative to sharing local data for vehicle privacy purposes, compression and bit reduction via quantization is essential. This proposed allocation scheme optimizes long-term reward in terms of the total training time and quantization error. Their work demonstrates the feasibility of using inexpensive devices for federated learning without requiring high-bandwidth infrastructure (Zhang et al., 2023).

While many previous research efforts primarily focused on task optimization, offloading techniques, and latency improvements, there remains a significant gap in the development of an openly accessible and low-cost vehicular edge computing platform for autonomous vehicles. The following chapters of this work propose such a platform, addressing this gap by providing an affordable, open-source solution that promotes collaboration and innovation in the autonomous vehicle research field.

METHODS

To address the challenges of cost and computational efficiency in autonomous vehicle systems, we designed a low-cost, open-source vehicular edge computing platform. The system integrates Donkey Cars, each built using the Waveshare PiRacer Pro AI Kit and equipped with a Raspberry Pi 4 and a Raspberry Pi Camera Module – a 5 MP, 160-degree FOV camera capable of capturing 720p video at 60 fps. The platform architecture leverages edge computing to enhance real-time processing while minimizing the computational burden on the vehicle itself. It consists of two key components: the Vehicle Unit (Client- Donkey Car) and Edge Computing Unit (Server- NVIDIA Jetson AGX Orin). Each Donkey Car can either execute a self-driving model locally or offload inference to the Jetson AGX Orin, for enhanced processing capabilities. Table 1 provides a detailed comparison of the computational capabilities of both devices, illustrating the significant processing advantage of the Jetson AGX Orin.

Table 1: Technical specification comparison between Raspberry Pi 4 and NVIDIA AGX Orin (14, 15).

Device	CPU	GPU
Raspberry Pi 4	Quad-core ARM Cortex-A72 CPU	VideoCore VI GPU
Jetson AGX Orin	12-core ARM Cortex-A78AE v8.2 64-bit CPU	Up to 2048-core Nvidia Ampere architecture GPU with 64 Tensor Cores

A dual-protocol communication architecture enables efficient interaction between a Donkey Car client and the edge server: UDP (User Datagram Protocol) is used for real-time image transmission from any Donkey Car client to the Jetson AGX Orin, ensuring low-latency data streaming, while TCP (Transmission Control Protocol) is used for sending processed control signals from the Orin module back to a Donkey Car client, ensuring reliable command execution. Both devices are also connected to the same WiFi network. Figure 1 illustrates this client-server interaction, where the vehicle streams real-time image data, the server processes it using machine learning models, and the computed control signals are transmitted back to the vehicle for navigation. Donkey Car clients transmit captured frames using the libcamera-vid software. These frames are captured at the default 30 frames per second setting at a resolution of 640 x 480.

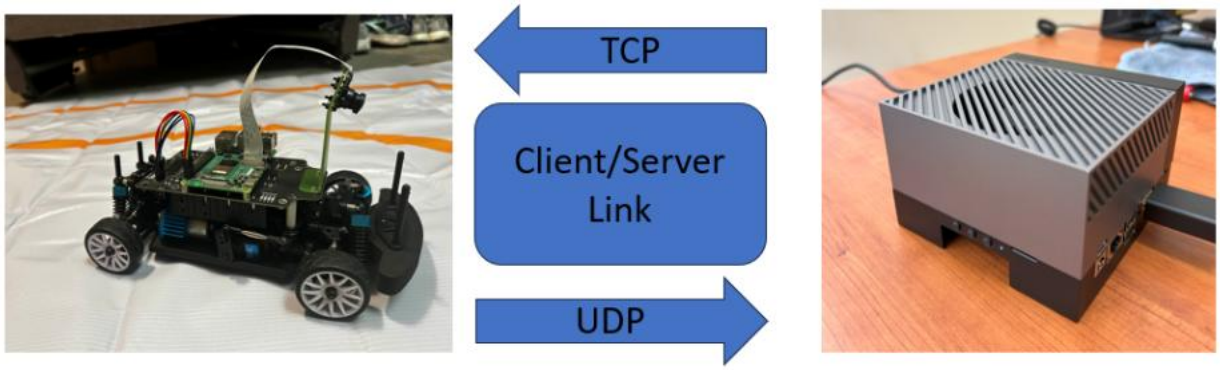

Figure 1: Client-server architecture of the Vehicle Edge Computing Platform, showing data flow between the Donkey Car and NVIDIA Jetson AGX Orin Module.

The platform is built on an open-source software stack, enabling flexible development and experimentation, which includes Operating Systems: Raspberry Pi OS (vehicle) and Ubuntu-based JetPack SDK (edge server); Machine Learning Framework: TensorFlow/Keras for neural network model execution; and Communication Implementation: Python-based UDP/TCP scripts for seamless data exchange.

EXPERIMENTS AND RESULTS

To evaluate the performance of our proposed vehicle edge computing platform, we conducted a series of controlled experiments with the Donkey Car-Jetson AGX Orin system. The primary objective was to compare the computational efficiency of local processing on the Raspberry Pi 4 against edge-based processing on the Orin module. Specifically, we measured CPU and memory usage, and inference latency, or otherwise known as inference time, under different machine learning models.

Computer Vision Models Used

For the experiments, we evaluated the performance of three different autopilot model architectures implemented using Keras and deployed with TensorFlowLite, an optimized framework for running machine learning models on mobile, embedded, and Internet of Things (IoT) devices with an emphasis on efficient inference. The three model types are referred to as: "linear", categorical, and recurrent neural network (RNN).

The "linear" model makes use of a single neuron following a series of convolutions to output the continuous values for steering and throttle via a Keras Dense layer with linear activation; thus, the "linear" designation refers only to the output layer. Meanwhile, the categorical model converts continuous steering and throttle into discrete bins, trains the network to predict a probability for each bin using categorical cross-entropy, and at runtime decodes that probability distribution into a chosen action. Finally, the RNN model converts a short sequence of recent images to features, which are then given to a recurrent block that maintains short-term memory, and the output from this block is decoded into steering and throttle values.

For each experimental trial, the Donkey Car(s) autonomously navigated a 3 by 2 meter oval track (Figure 2) for 60 seconds. During these trials, we collected performance metrics for both local and edge-based inference. To monitor system performance, we used the psutil library on the model-running device to track CPU and memory usage. These metrics were collected for every frame: CPU usage reflects the average CPU utilization across all cores, while memory usage measures the memory consumption of the Python-running process. Inference latency was measured using the Python time module, which recorded the duration of each model inference.

Figure 2: Testbed setup, Donkey Car placed on oval 3 by 2 meter track

Although message transmission time between the Donkey Car and Orin module can vary depending on several factors, including, network speed and congestion, we made an effort to evaluate communication overhead by attaching timestamps to messages sent between the Orin module and the Donkey Car, providing a useful approximation of communication delay and helped us quantify impact of streaming video frames to the Jetson AGX Orin, as well as assess the overall effectiveness of our client-server architecture. The difference between the received and current timestamps measured provides the estimated transmission time.

Table 2 presents communication latency metrics measured from our private WiFi network powered by the RT-AC68R Dual Band ASUS Gigabit Router. The table shows the latency for video frame transmission from the Raspberry Pi Donkey Car to the Orin module via the UDP protocol, as well as the latency for control signal transmission from the Orin module back to the Donkey Car via the TCP protocol. These latency measurements were collected during trials where the model was run on the Orin device. The results indicate that latency in both directions and protocols is minimal.

Table 2: Average median UDP and TCP transmission sample times after running six trials for each sample size, measured from our private network.

Orin to Donkey Car (UDP Transmission)		Donkey Car to Orin (TCP Transmission)	
Sample Size	**Avg. Median**	**Sample Size**	**Avg. Median**
300	**1.72**	**300**	**5.27**
500	**1.56**	**500**	**5.11**

For the initial single-vehicle experiment, a single-threaded program is used on the Orin device to receive the frames, process them for steering and throttle values, and to send the values back to the Donkey Car, while also gathering metrics on average CPU usage, memory usage, and inference latency.

The results, summarized in Table 3, demonstrate that using the NVIDIA Jetson AGX Orin leads to a significant reduction in average CPU usage, memory usage, and inference latency compared to local processing on the Raspberry Pi 4. The "linear", categorical, and RNN models will hereafter be referred to as TensorFlow Linear, TensorFlow Categorical, and TensorFlow RNN, respectively. Specifically, there is a 99.7% reduction in inference latency, a 67.0% reduction in CPU usage, and a 69.3% reduction in memory utilization with Tensorflow Linear. Similar improvements are observed with TensorFlow Categorical, where CPU usage is reduced by 66.1%, memory usage by 68.4%, and inference latency by 99.8%. Performance metrics from the RNN trials show even more pronounced differences, with an 80.4% reduction in CPU usage, a 54.1% reduction in memory usage, and a 99.7% reduction in inference latency.

Table 3: Single-vehicle, single-threaded performance comparison of TensorFlow models on Raspberry Pi 4 and NVIDIA AGX Orin.

Modality		**Local (Raspberry Pi 4)**					**Edge (NVIDIA AGX Orin)**					
Models	Metrics	Trial 1	Trial 2	Trial 3	Trial 4	**Avg**	Trial 1	Trial 2	Trial 3	Trial 4	**Avg**	**Improve**
TensorFlow Linear	Avg CPU Usage (%)	34.1	38.2	34.9	33.1	**35.1**	10.9	10.9	13.6	10.9	**11.6**	67.0%
	Avg Memory Usage (%)	13.6	13.9	13.7	13.6	**13.7**	4.2	4.2	4.4	4.2	**4.2**	69.3%
	Avg Inference Time (ms)	1570.9	1604.5	1528.5	1678.0	**1595.5**	4.66	4.66	4.55	4.66	**4.6**	99.7%
TensorFlow Categorical	Avg CPU Usage (%)	30.2	29.8	29.4	29.8	**29.8**	10.2	10.2	9.9	10.2	**10.1**	66.1%
	Avg Memory Usage (%)	13.6	13.5	13.6	13.6	**13.6**	4.2	4.3	4.3	4.3	**4.3**	68.4%
	Avg Inference Time (ms)	1511.8	1599.2	1500.1	1535.8	**1536.7**	3.7	3.7	3.7	3.7	**3.7**	99.8%
TensorFlow RNN	Avg CPU Usage (%)	58.7	59.0	54.6	58.7	**57.8**	11.0	11.5	11.6	11.0	**11.3**	80.4%
	Avg Memory Usage (%)	14.2	14.1	14.1	14.1	**14.1**	6.5	6.5	6.5	6.4	**6.5**	54.1%
	Avg Inference Time (ms)	1559.3	1551.9	1503.4	1476.4	**1522.7**	4.7	4.7	4.7	4.7	**4.7**	99.7%

The next performance test conducted involved two vehicles, conducted to evaluate the Jetson AGX Orin's ability to manage two Donkey Cars simultaneously, given that a real-world scenario would involve multiple vehicles. To support this setup, a multi-threaded program was developed for efficient communication and processing. Specifically:

- Two threads per Donkey Car: one to receive frames and another to send steering and throttle commands.
- A minimum of four threads were dedicated to processing frames through the self-driving model.
- One thread continuously monitors and reports connection status.
- Another thread records and reports performance metrics.

This thread-based architecture enabled parallel handling of the two vehicles and tasks, allowing us to assess the scalability of the system with an additional vehicle. Queues and locks are used to prevent concurrency issues.

Table 4 demonstrates the performance results of the two-vehicle, multi-threaded experiment, evaluated using the same metrics as before. The metrics collection process from the previous experiment was also reused. It is important to note that this experiment does not include the TensorFlow RNN model from the previous experiment, as integrating this model into the

multithreaded program required significantly more development time. Incorporating this model can be left for future work.

The performance results demonstrate that average CPU usage for both models is nearly double that of the single-threaded experiment, coming at 21.67% and 21.47% for linear and categorical models, respectively. Average memory usage is also down significantly, dropping to 0.8%. Finally, average inference time is doubled from the single-threaded experiment for the same two models, doubling to 9.67 milliseconds for both kinds of models.

Table 4: Two-vehicle, multi-threaded performance comparison of TensorFlow models on NVIDIA AGX Orin using CPU

Models	**Metrics**	Trial 1	Trial 2	Trial 3	**Avg**
TensorFlow Linear	Avg CPU Usage (%)	21.7	21.9	21.4	**21.67**
	Avg Memory Usage (%)	0.8	0.8	0.8	**0.8**
	Avg Inference Time (ms)	9.7	9.7	9.6	**9.67**
TensorFlow Categorical	Avg CPU Usage (%)	22.0	21.2	21.2	**21.47**
	Avg Memory Usage (%)	0.8	0.8	0.8	**0.8**
	Avg Inference Time (ms)	10.0	10.0	9.0	**9.67**

VEHICLE EDGE COMPUTING FOR TRAVEL BEHAVIOR AND DEMAND

Understanding and modeling future travel behavior and demand requires high-resolution, time-sensitive observations of how vehicles interact with their environment and with one another. Traditional approaches to travel behavior analysis often rely on aggregated traffic sensors, probe vehicle data processed in the cloud, or post hoc analysis of logged trajectories. While these methods have proven effective at scale, they are limited in their ability to capture fine-grained behavioral responses, such as instantaneous reactions to obstacles, lane positioning decisions, or short-term speed adjustments, particularly under strict latency, bandwidth, and privacy constraints (Olariu, 2020; Liu et al., 2019). Vehicle edge computing provides a complementary paradigm by enabling real-time, localized processing of perceptual and control data at or near the point of data generation (Shi et al., 2016).

In this project, the proposed vehicle edge computing platform demonstrates how edge-based perception and inference can serve as an enabling layer for future travel behavior and demand research. By offloading real-time perception and control tasks from vehicles to nearby edge servers, the platform supports millisecond-level inference while reducing computational burden on onboard hardware. Although the experiments in this report focused on system performance and scalability, the resulting architecture directly facilitates the collection and interpretation of behavioral signals that are central to travel demand analysis and intelligent transportation systems (Liu et al., 2019; Lu & Shi, 2023).

Edge-based perception systems, such as those evaluated in this work, can extract detailed vehicle motion characteristics from camera data, including speed profiles, acceleration and deceleration patterns, lane positioning, and following behavior. These low-level motion features are fundamental building blocks of travel behavior models, as they reflect driver or vehicle decision-making in response to infrastructure, traffic conditions, and external stimuli (Anderson et al., 2016). When processed at the edge, these features can be captured with minimal latency, preserving the temporal relationship between environmental inputs and vehicle responses. This is particularly important for studying behavior in dynamic settings, such as intersections, merging zones, or shared spaces with pedestrians and cyclists, where delayed inference can obscure causal relationships.

Compared to cloud-centric processing pipelines, edge computing offers distinct advantages for behavior-focused applications. High communication latency and bandwidth constraints in cloud systems can blur short-term behavioral responses or necessitate aggressive data downsampling (Olariu, 2020). In contrast, localized edge processing enables event-driven analysis, where behavioral changes can be detected and logged at the moment they occur. The low inference latencies demonstrated in this project align with prior findings that emphasize the suitability of edge computing for time-critical vehicular applications (Shi et al., 2016; Liu et al., 2019).

Roadside edge nodes also play a critical role in bridging individual vehicle behavior and aggregate travel demand. By serving as local aggregation points, edge servers deployed at infrastructure locations (e.g., intersections or corridors) can combine observations from multiple vehicles to estimate local traffic states, such as queue formation, flow rates, and short-horizon demand fluctuations. These localized estimates can support adaptive traffic management strategies and inform demand-responsive control systems without requiring all raw data to be transmitted to centralized cloud servers (Fan et al., 2023; Yin et al., 2024). The multi-vehicle experiments presented in this report illustrate the feasibility of concurrently supporting multiple vehicles on a single edge node, an important step toward scalable, demand-aware transportation systems.

In addition to latency and scalability benefits, vehicle edge computing offers meaningful advantages for privacy-preserving data collection. Rather than storing or transmitting raw video streams, behavioral features can be extracted locally and shared in abstracted form, reducing the risk of exposing personally identifiable information. This approach aligns with prior research advocating edge-based processing as a mechanism for balancing data utility with privacy in vehicular networks (Zhang et al., 2024). Furthermore, the low-cost and open-source nature of the proposed platform lowers barriers to entry for academic institutions and public agencies, facilitating equitable access to advanced data collection capabilities for travel behavior research.

The use of Donkey Cars as experimental platforms further supports this research direction by enabling controlled, reproducible studies of vehicle behavior and interaction. While small in scale, these platforms allow rapid prototyping and validation of edge-based inference pipelines before deployment in full-scale transportation systems. Such testbeds are widely recognized as valuable tools for early-stage intelligent transportation research, particularly when evaluating new computing architectures and communication paradigms (Lu & Shi, 2023).

It is important to note that this project does not directly estimate travel demand or develop behavioral models. Instead, it establishes a technical foundation upon which such analyses can be built. By demonstrating a low-latency, scalable, and accessible vehicle edge computing platform, this work enables future research that integrates behavioral modeling, demand estimation, and intelligent transportation systems. In this sense, vehicle edge computing serves as a critical enabler for advancing the understanding of future travel behavior and demand in connected and automated transportation environments (Shi et al., 2016; Liu et al., 2019).

POSSIBLE AVENUES FOR FUTURE WORK

There are numerous opportunities for future research using our vehicle edge computing platform. A natural next step is to incorporate additional Donkey Cars into our platform and to expand to a larger, more complex track. However, we encountered limitations in this direction: some Donkey Cars available to us suffered from mechanical and camera reliability issues, and our physical space constrained the scale of track expansion. Implementing Cellular Vehicle-to-Everything (C-V2X) communication technology in place of the UDP/TCP communication link would also be a worthwhile endeavor to explore if that technology becomes available and affordable.

We also began developing an experiment to evaluate the AGX Orin's GPU performance for real-time inference while communicating with multiple Donkey Cars simultaneously. This line of investigation remains promising for understanding the limits of on-board edge computation under multi-vehicle workloads.

Another avenue for future exploration involves integrating CHI@Edge—a University of Chicago testbed that provides support and hardware for edge computing experiments—into our platform as an alternative to the Orin module (Chameleon Cloud, 2025). CHI@Edge offers leased access to edge nodes within a private virtual network, allowing users to run applications on specific edge devices. In our case, only Jetson Nano and Jetson Xavier NX devices were accessible through CHI@EDGE. A key distinction of CHI@Edge is that it provides cloud-like, remotely managed access to distributed edge devices, whereas the AGX Orin represents a self-contained, local edge computing unit. Future work should further investigate the trade-offs between this cloud-managed model and our localized solutions such as the Orin or the Raspberry Pi.

Future research may also incorporate more advanced perception models into the platform. Candidate models include the You Only Look Once (YOLO) series of single-stage object detectors optimized for speed (Nelson, 2025), and Meta's Detectron2 computer vision library (Wu et al., 2019), which offers state-of-the-art detection and segmentation capabilities.

Beyond expanding the Donkey Car platform, future work could explore more sophisticated autonomous vehicle systems, e.g. the Quanser Car, which employs a richer sensor suite than monocular cameras alone. These systems may come equipped with multiple cameras, radar, ultrasonic sensors, inertial measurement units, and inexpensive forms of LiDAR, particularly 2D LiDAR, offering enhanced environmental perception, improved robustness in adverse conditions, and more accurate localization and obstacle detection capabilities.

CONCLUSIONS AND POLICY IMPLICATIONS

In this project, we presented the design and implementation of a low-cost, open-source vehicle edge computing platform aimed at enhancing the computational capabilities of autonomous vehicles while reducing reliance on expensive onboard hardware. By leveraging a client-server architecture, where a Donkey Car equipped with a Raspberry Pi 4 offloads computations to an NVIDIA Jetson AGX Orin module, our platform significantly improves processing efficiency, reduces inference latency, and lowers CPU and memory usage. The experimental results demonstrate the feasibility of deploying edge computing solutions to achieve real-time decision-making and hazard detection in autonomous vehicles, even with limited onboard resources.

For future work, the following is suggested:

- Integrating more complex track layouts, multiple vehicles, and C-V2X technology
- Enabling cooperative interactions between multiple clients
- Incorporating GPU acceleration for inference tasks
- Comparing the performance of CHI@EDGE with that of a local Jetson AGX Orin
- Integrating and evaluating advanced perception models such as YOLO and Detectron2 into the platform

The development of accessible and scalable vehicle edge computing platforms has several implications for public policy, regulatory frameworks, and urban planning. First, low-cost, open-source solutions can lower barriers to entry for small businesses, startups, and academic institutions seeking to participate in autonomous vehicle research and deployment, fostering innovation while reducing dependence on proprietary, high-cost hardware. Policymakers could incentivize the adoption of such platforms through grants, research subsidies, or inclusion in smart city initiatives.

Second, the demonstrated latency reductions and efficiency gains highlight the potential for safer and more reliable autonomous driving systems. Regulators can use these findings to guide standards for real-time processing and communication requirements in autonomous vehicles, ensuring that both private and public fleets meet minimum safety and performance benchmarks. This is particularly relevant for connected vehicle infrastructure, including roadside units and edge servers, where local processing can complement cloud-based systems to enhance traffic management and hazard mitigation.

Third, by showing that multi-vehicle coordination is feasible on low-cost hardware, this work suggests a path for policies promoting cooperative and connected vehicle networks. Urban planners and transportation authorities could integrate vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication standards into traffic management systems, enabling real-time adaptive control of congestion, emergency response, and autonomous fleet operations.

Finally, the open-source nature of the platform supports transparent, reproducible, and collaborative research, which can inform evidence-based policy decisions. Regulatory agencies can leverage such accessible testbeds to evaluate autonomous vehicle safety, conduct pilot programs, and develop guidelines for ethical AI usage, data privacy, and cybersecurity in transportation networks. By encouraging public-private partnerships that adopt these platforms,

governments can accelerate the deployment of intelligent transportation systems while ensuring fair access to the benefits of autonomous technology.

By bridging the gap between affordability and computational efficiency, this platform provides a foundation for further research in vehicle edge computing at low cost. Ultimately, such systems have the potential to accelerate the development of intelligent transportation solutions, making autonomous technologies more accessible, scalable, and adaptable to a wide range of mobility needs, while informing policy that balances innovation, safety, and public interest.

REFERENCES

Anderson, J. M., Kalra, N., Stanley, K. D., Sorensen, P., Samaras, C., & Oluwatola, O. A. (2016). *Autonomous vehicle technology: A guide for policymakers*. RAND Corporation. https://doi.org/10.7249/RR443-2.

Automotive Edge Computing Consortium. (2021). *Distributed computing in an AECC system* (Version 1.0.0) [White paper]. AECC.

Bojarski, M., Testa, D. D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., & Zieba, K. (2016). *End to end learning for self-driving cars* (arXiv:1604.07316). arXiv. https://arxiv.org/abs/1604.07316

Chameleon Cloud. (n.d.). *CHI@Edge*. https://www.chameleoncloud.org/experiment/chiedge/

Deloitte. (2023). *Future of cloud survey: Industry cloud trends* [Report]. Retrieved January 7, 2025, from https://www.deloitte.com/us/en/services/consulting/content/cloud-strategy-innovation-survey-report.html

Fan, W., Su, Y., Liu, J., Li, S., Huang, W., Wu, F., & Liu, Y. (2023). Joint task offloading and resource allocation for vehicular edge computing based on V2I and V2V modes. *IEEE Transactions on Intelligent Transportation Systems, 24*(4), 4277–4292.

Feng, W., Zhang, N., Li, S., Lin, S., Ning, R., Yang, S., & Gao, Y. (2022). Latency minimization of reverse offloading in vehicular edge computing. *IEEE Transactions on Vehicular Technology, 71*(5), 5343–5357.

Liu, S., Liu, L., Tang, J., Yu, B., Wang, Y., & Shi, W. (2019). Edge computing for autonomous driving: Opportunities and challenges. *Proceedings of the IEEE, 107*(8), 1697–1716.

Lu, S., & Shi, W. (2023). Vehicle computing: Vision and challenges. *Journal of Information and Intelligence, 1*(1), 23–35.

Mallozzi, P., Pelliccione, P., Knauss, A., Berger, C., & Mohammadiha, N. (2019). Autonomous vehicles: State of the art, future trends, and challenges. In *Software engineering for resilient systems* (pp. 347–367). Springer International Publishing.

Nelson, J. (2025). What is YOLO? The ultimate guide. Roboflow Blog. https://blog.roboflow.com/guide-to-yolo-models/

NVIDIA. (n.d.). *NVIDIA Jetson Orin*. https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/

Olariu, S. (2020). A survey of vehicular cloud research: Trends, applications, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 21(6), 2648–2663.

Raspberry Pi. (n.d.). *Raspberry Pi 4 tech specs*.

https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal, 3*(5), 637–646.

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019). Detectron2 [Computer Software]. https://github.com/facebookresearch/detectron2

Yin, L., Luo, J., Qiu, C., Wang, C., & Qiao, Y. (2024). Joint task offloading and resources allocation for hybrid vehicle edge computing systems. *IEEE Transactions on Intelligent Transportation Systems, 25*(8), 10355–10368.

Zhang, C., Zhang, W., Wu, Q., Fan, P., Fan, Q., Wang, J., & Letaief, K. B. (2024). *Distributed deep reinforcement learning based gradient quantization for federated learning enabled vehicle edge computing* (arXiv:2407.08462). arXiv. https://arxiv.org/abs/2407.08462