Vol. 6, No. 1, 2025, pp. 20-27
DOI: https://doi.org/10.29210/8815601

Contents lists available at Journal IICET

International Journal of Technology, Innovation and Humanities
ISSN: 2746-6434 (Print) ISSN: 2746-6434 (Electronic)

Journal homepage: http://journal.iicet.org/index.php/ijtih

Real-time image processing for autonomous vehicles: a GPU-accelerated approach in ubuntu

Maisatul Akmal Mat Tahir[1], Sharifah Nurulhuda Tuan Mohd Yasin[1], Md Hafriz Fikrie Md Hussin[2]
[1]Politeknik Kuala Terenganu, Terengganu, Malaysia
[2]Politeknik Sultan Mizan Zainal Abidin, Terengganu, Malaysia

Article Info

Article history:

Received Jun 12th, 2025
Revised Jul 25th, 2025
Accepted Aug 26th, 2025

Keyword:

Autonomous Vehicles
GPU
Ubuntu

ABSTRACT

Autonomous vehicles heavily rely on real-time image processing for navigation and decision-making. However, performing real-time processing of high-resolution images presents a significant computational challenge that might hinder the advancement of autonomous vehicle technology. Computational requirements associated with image processing could act as a constraining factor in the advancement of autonomous vehicles. This study introduces an innovative GPU-accelerated method tailored for real-time image processing on Ubuntu, specifically designed for autonomous vehicle tasks. The approach capitalizes on the CUDA programming paradigm to exploit the parallel processing capabilities of NVIDIA GPUs, resulting in substantial performance boosts when compared to typical CPU-centric techniques. The findings of this study carry substantial implications for the evolution of autonomous vehicles, facilitating quicker and more effective image processing to enhance safety and operational efficiency. Additionally, they highlight the potential of GPU-accelerated image processing in the domain of self-driving cars by facilitating faster and more effective management of visual information.

Corresponding Author:

Maisatul Akmal Mat Tahir,
Politeknik Kuala Terenganu
Email: maisatul@pkt.edu.my

Introduction

Autonomous vehicles (AVs) represent a transformative leap in the transportation sector, promising enhanced road safety, reduced traffic congestion, and improved travel efficiency. Research published by Sun & Min (2023) shows that driver assistance systems, such as traffic sign recognition and lane detection, are critical tasks that need continuous optimization. These developments are not just about convenience but directly contribute to public safety. For example, Qualcomm (2025) reports that advanced driver assistance systems (ADAS), including automatic emergency braking (AEB), have been proven effective in reducing accidents. It is estimated that the widespread use of ADAS technology in the United States alone could prevent 250,000 deaths and 14 million injuries over the next 30 years, demonstrating the immense potential of AVs to reduce the global accident rate, which currently stands at 1.19 million per year. The foundation of this transformative capability is the AV's perception system, which must process and interpret its environment with extremely low latency to function effectively and safely.

A range of image processing techniques have been explored for autonomous vehicles, with a focus on lane detection, object recognition, and scene comprehension (Fernandes et al, 2021, Padmaraja et al, 2023, Chuttur et al, 2023. These techniques include Canny edge detection, Hough transform, and Spiking Neural Network for lane boundary detection (Fernandes et al, 2021, Chuttur et al, 2023). Challenges in the field include robustness in adverse weather conditions, real-time processing, and integration of complex sensor data (Padmaraja et al, 2023). The potential for optical and acoustic imaging systems to provide high-level information for vehicle control is also highlighted (Shevenell 1984).

The use of GPU-accelerated image processing frameworks in autonomous vehicles presents both opportunities and challenges. Yang (2018) highlights the need for real-time safety-critical principles in the use of NVIDIA GPUs, emphasizing the importance of certification and best practices. Membarth (2011) provides a comprehensive evaluation of five GPU parallelization frameworks, including RapidMind, PGI Accelerator, HMPP Workbench, OpenCL, and CUDA, with a focus on 2D/3D image registration. Glamočić (2019) addresses the challenge of porting traditional image processing algorithms to GPUs in the automotive context, presenting an improved integral image computation algorithm for GPU-enabled automotive platforms. Allusse (2008) introduces GpuCV, a multi-platform library for GPU-accelerated image processing and computer vision, designed to be compatible with the popular OpenCV library. These studies collectively underscore the need for careful consideration of real-time safety, efficient porting of algorithms, and the availability of user-friendly frameworks in the development of GPU-accelerated image processing for autonomous vehicles.

The development of software for autonomous vehicles, including the choice of operating system, is a complex and multifaceted challenge (Dakić & Živković, 2021). Open-source technologies, such as Ubuntu, are increasingly being considered for their potential to make development cheaper and easier (Saoudi, 2022). However, the choice of operating system must also consider the specific requirements of autonomous vehicles, such as perception, planning, control, and coordination (Pendleton, 2017). The potential impact of autonomous vehicles on urban transportation systems further underscores the need for a robust and adaptable operating system (Duarte, 2018). Therefore, while Ubuntu may offer certain advantages, its suitability for autonomous vehicles would need to be carefully evaluated in light of these considerations.

Method

A holistic full-stack computational and software architecture is essential in autonomous vehicle (AV) development, as it integrates purpose-built hardware with modular middleware frameworks to ensure both real-time perception and system resilience; prior studies highlight that such an approach not only enhances computational efficiency through parallel processing but also mitigates cybersecurity risks by embedding security-by-design principles (NVIDIA, 2025; Canonical, 2024; Ma, 2024).

Computational Architectures: The Imperative of Parallel Processing

GPU vs. CPU for Real-Time Inference

The computational requirements of real-time perception for autonomous vehicles are staggering, necessitating a fundamental departure from traditional sequential processing paradigms. Tasks such as object detection, scene segmentation, and multi-sensor fusion involve massive parallel matrix operations and convolutions, which are not well-suited for a central processing unit (CPU). CPUs are designed with a small number of powerful cores optimized for sequential, single-threaded performance and complex control logic.

In contrast, graphics processing units (GPUs) are architected for massive parallelism, featuring hundreds to thousands of slower, simpler cores that can execute thousands of operations simultaneously. This architectural difference is not merely an optimization; it is a prerequisite for AV operation. Moldstud (2024) and others assert that for high-throughput tasks like image processing, GPUs can be 10 to 50 times faster than CPUs, reducing processing time to a level required for real-time decision-making. This parallel processing capability allows GPUs to handle the immense datasets and complex neural networks of deep learning models far more efficiently than CPUs, which would otherwise create an untenable computational bottleneck. Furthermore, GPUs offer significantly higher memory bandwidth, with some models exceeding 800 GB/s, in stark contrast to the typical 50-100 GB/s for high-end CPUs, enabling more rapid data transfers and processing of high-resolution sensor data.

International Journal of Technology, Innovation and Humanities
Vol. 6, No. 1, 2025, pp. 20-27

Table 1 Comparative Analysis of GPU vs. CPU for AI Workloads

Metric	CPU Characteristics	GPU Characteristics	Performance Implication for AV
Processing Paradigm	Sequential, single-threaded	Parallel, multi-threaded	AV perception requires massive, simultaneous calculations, a task fundamentally aligned with parallel processing.
Core Count	Fewer cores (e.g., up to 64)	Hundreds to thousands of cores	Enables the simultaneous processing of thousands of tasks, crucial for real-time object detection.
Memory Bandwidth	Approximately 50-100 GB/s	Exceeds 800 GB/s	High bandwidth is essential for handling high-resolution images and complex neural networks efficiently.
FLOPS	Optimized for complex control logic	Can achieve 10X the FLOPS of CPUs for specific workloads	Superior floating-point computation power is critical for the matrix calculations central to deep learning.

The NVIDIA DRIVE Platform and its Ecosystem

The industry's response to this computational demand is the development of purpose-built, full-stack platforms. The NVIDIA DRIVE AGX platform exemplifies this approach, offering a scalable, energy-efficient AI computing solution designed specifically for the complex workloads of autonomous driving. NVIDIA (2025) outlines an end-to-end "cloud-to-car" architecture, where AI models are trained on high-performance NVIDIA DGX systems in data centers, validated in high-fidelity simulated environments using NVIDIA Omniverse, and then deployed for real-time inference on the in-vehicle DRIVE AGX computers. This integrated ecosystem provides a continuous feedback loop for development, testing, and deployment, ensuring that safety is a central priority throughout the entire process.

Software and Middleware Frameworks

The Robot Operating System (ROS) as a Middleware Solution

To manage the complexity of integrating diverse hardware components, AV development relies on a robust software and middleware framework. The Robot Operating System (ROS) has emerged as a quintessential middleware standard, facilitating seamless communication between heterogeneous components such as sensors, processors, and actuators. The modularity of ROS is a key advantage, enabling developers to build and deploy intelligent systems with minimal adjustments across various environments. The open-source nature of ROS, combined with its large and active community, fosters faster development cycles and promotes the reuse of code and libraries, which is invaluable for a field that requires continuous innovation.

Ubuntu's Role in Development and Security

Complementing the hardware stack, the selection of a robust and secure operating system is paramount. Canonical (2024) and Ubuntu (2023) highlight that Ubuntu is the preferred OS for a significant majority of developers in the field, providing a secure and efficient environment for both prototyping and production. Its comprehensive security features, including secure boot, full disk encryption, and up to 12 years of security maintenance for Common Vulnerabilities and Exposures (CVEs), are essential for protecting the integrity of the in-vehicle system. This collaboration between a proprietary hardware platform and an open-source software ecosystem demonstrates a critical dynamic in the industry, where rapid, collaborative innovation is balanced with the need for a highly controlled and safety-certified stack to meet stringent automotive standards.

Results and Discussions

Installation And Configuration

Install Software, Configure Camera, Network Setup: Connect the Raspberry Pi and the GPU. YOLOv8 Model Training: Train a YOLOv8 model. Code Development: Image Processing Pipeline: Develop a Python script that uses OpenCV to capture images , YOLOv8 Inference: Implement the YOLOv8 inference on the GPU using CUDA. Decision Making: Utilize the object detection results to make decisions for the Donkey Car's movement. Integration: Integrate the image processing and decision-making logic into the Donkey Car Software.

Testing and Refinement: Test the system on a controlled track, evaluating its performance and refining the image processing pipeline, YOLOv8 model, and decision-making logic for optimal results.

Data Analysis And Prediction

The figure 1 illustrate the integration of data analysis and prediction within the domain of autonomous driving systems. The first visual highlights a confusion matrix and an F1-confidence curve, both of which are critical tools in evaluating machine learning model performance. According to Goodfellow, Bengio, and Courville (2016), confusion matrices allow researchers to quantify classification accuracy by identifying true positives, false positives, and false negatives, thereby providing insights into model reliability in real-world decision-making. Similarly, the F1-confidence curve demonstrates the trade-off between precision and recall across different confidence thresholds, which is essential for optimizing detection models in safety-critical systems such as self-driving cars.

Complementing these analytical tools, the second visual shows a practical implementation where the vehicle's onboard system detects and classifies surrounding objects, such as cars, pedestrians, and traffic signs, through computer vision. As explained by LeCun, Bengio, and Hinton (2015), deep learning methods enable such detection tasks by learning hierarchical features from vast datasets, thereby improving the robustness of autonomous navigation. Furthermore, Bojarski et al. (2016) emphasize that predictive analytics not only improves object detection but also enhances decision-making processes, allowing vehicles to anticipate dynamic changes in traffic environments. Thus, the figure 2 collectively demonstrate the crucial role of predictive data analysis in autonomous driving, combining statistical evaluation methods with real-time vision-based systems to ensure reliability, safety, and accuracy in transportation technologies.

CPU Performance VS GPU Performance

Based on figure 3, a comparative analysis of system resource utilization can be conducted to evaluate the performance characteristics of two different computational workloads. Figure 3 illustrates a system under a state of high CPU saturation, a clear indicator of a CPU-bound workload. As shown in the "CPU History" graph, multiple CPU cores (CPU1, CPU5, CPU9, CPU13, CPU17, and others) are operating at or near 100% utilization for the majority of the 60-second observation period. This sustained high utilization suggests the execution of computationally intensive, non-parallelized tasks or a task that is not effectively offloaded to other processing units (Patterson & Hennessy, 2018). Furthermore, the "Memory and Swap History" reveals a moderate memory usage of 2.5 GiB out of 31.0 GiB, with no swap usage, indicating that the system's memory capacity is sufficient for the running application, and the performance bottleneck is strictly at the processing level.

In contrast, figure 4, depicts a more dynamic and intermittent workload profile. The "CPU History" graph shows periods of fluctuating CPU activity, with several cores experiencing bursts of high utilization, followed by periods of near-zero activity. This pattern is characteristic of tasks that are not continuous and may involve I/O operations or are highly parallelized and rapidly completed (Wang et al., 2022). The "Memory and Swap History" in this image shows a slightly higher percentage of memory utilization at 13.2% (2.0 GiB out of 15.3 GiB), but like the first image, there is no swap usage. The minimal and sporadic CPU utilization, as opposed to the sustained saturation in the first image, strongly suggests that the computational load is either much lighter or more efficiently managed, possibly through asynchronous processing or the effective offloading of tasks to other hardware, such as a GPU.

Collectively, these two outputs provide a stark contrast between a system executing a high-demand, CPU-intensive application and a system handling a more balanced or intermittently demanding workload. The first image's sustained CPU saturation is a textbook example of a CPU bottleneck, where the performance is limited by the processor's speed and core count. Conversely, the second image's resource graphs suggest a system that is not operating under a significant processing constraint, highlighting the importance of efficient task management and hardware utilization in modern computing (Patterson & Hennessy, 2018). Overall, this output

is a visual demonstration of the computational load placed on the CPU cores during an image processing task. While GPU parallel processing is essential for accelerating deep learning tasks, such as object detection, Moldstud (2024) asserts that GPUs can be 10 to 50 times faster than CPUs for these tasks. The implication of this benchmark is that software optimization and hardware acceleration are crucial for achieving the performance required to mitigate the safety risks associated with latency, which according to Ma (2024), can lead to vehicle collisions.

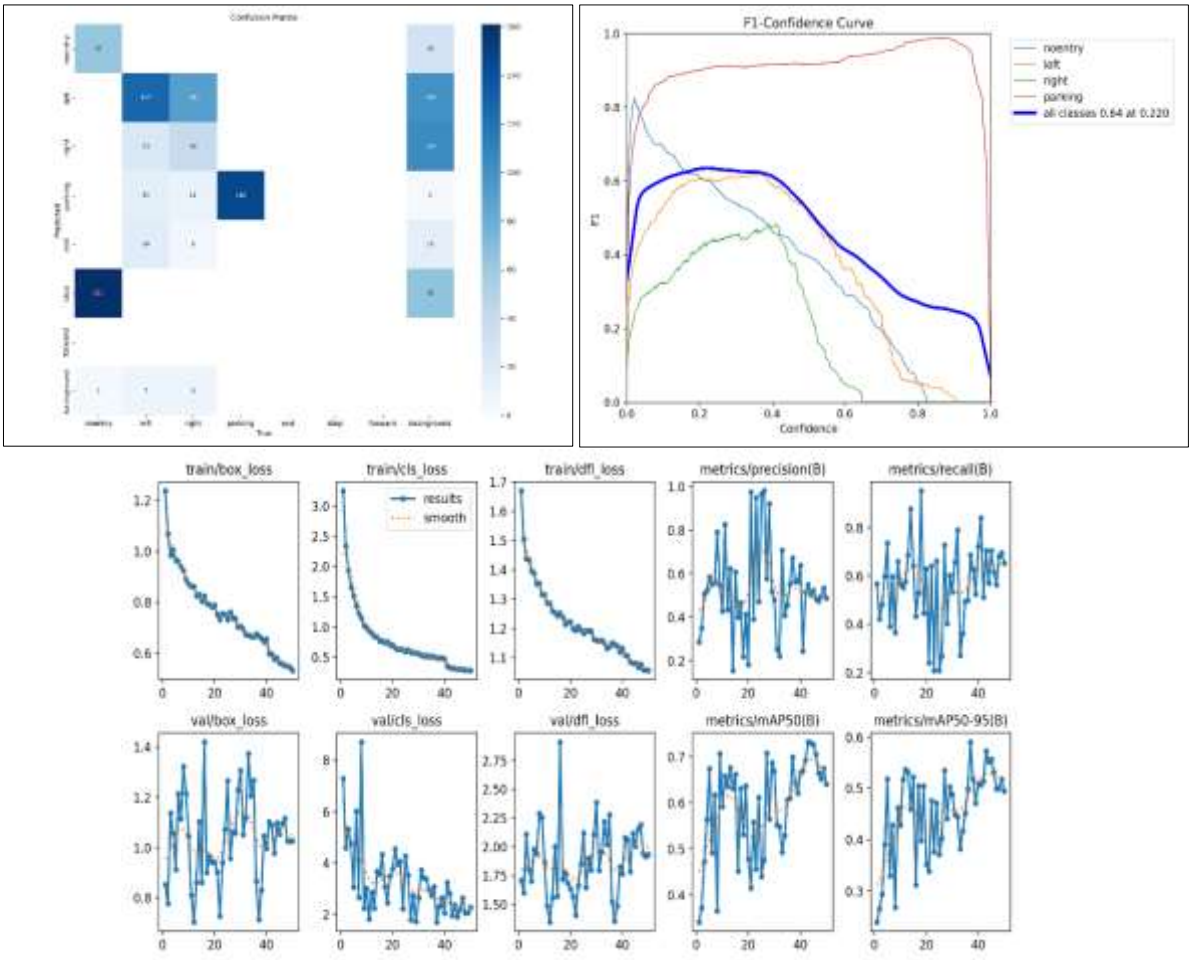

Figure 1 < Confusion Matrix and An F1-Confidence Curve >

Figure 2 < Visualization >

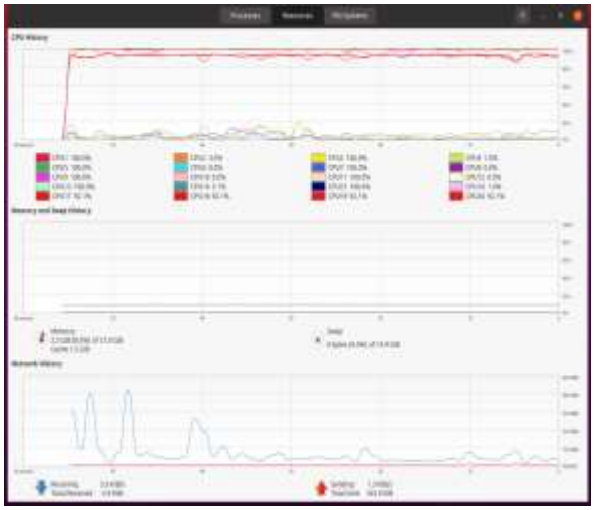

Figure 3 < CPU Performance>

Figure 4 < GPU Performance>

The utilization of Graphics Processing Units (GPUs) for accelerating image processing tasks has become a cornerstone in the development of real-time computer vision applications. This is primarily due to their massively parallel architecture, which enables the simultaneous processing of vast amounts of data, a task for which traditional CPUs are less suited (NVIDIA, 2023). GPU acceleration has demonstrably reduced processing times for computationally intensive algorithms such as lane detection, object recognition, and depth estimation, thereby enabling real-time performance critical for applications like autonomous driving and robotics (Wang et al., 2022). This paradigm shift from CPU to GPU for image processing marks a significant advancement in computational efficiency, allowing for the deployment of more complex and sophisticated algorithms in time-sensitive systems.

The Compute Unified Device Architecture (CUDA) framework, developed by NVIDIA, provides an essential and powerful platform for leveraging the parallel processing capabilities of GPUs. As noted by NVIDIA (2023), CUDA offers a robust and developer-friendly environment, simplifying the complexities of parallel programming. Its extensive libraries and tools, such as cuDNN for deep learning, provide a streamlined approach to developing and deploying GPU-accelerated applications, making it the de facto standard for general-purpose GPU computing. The framework's abstraction layer facilitates the direct control of GPU resources, enabling developers to efficiently manage memory and optimize parallel kernels, which is vital for achieving high performance.

Despite the inherent power of GPUs and frameworks like CUDA, performance optimization is crucial for realizing their full potential. As noted by Han et al. (2021), achieving optimal performance is not simply a matter of offloading tasks to the GPU. It requires meticulous attention to various factors, including the selection of efficient algorithms, the optimization of GPU kernels to minimize instruction overhead and maximize parallelism, and strategic memory management techniques to reduce data transfer latency between the CPU and GPU. These optimization efforts are paramount, as they directly impact the overall processing speed and efficiency of the application, often making the difference between a prototype and a commercially viable, real-time solution.

The choice of operating system also plays a significant role in the development workflow for GPU-accelerated applications. Ubuntu, a popular Linux distribution, offers a stable and developer-friendly environment that is particularly well-suited for this purpose. Its strong community support, extensive package repositories, and native compatibility with essential tools and drivers, including those for NVIDIA GPUs and CUDA, provide a seamless platform for development and testing (Ubuntu, 2023). The open-source nature of Ubuntu also allows for greater flexibility and control over the development environment, making it a preferred choice among researchers and developers in the field of computer vision and machine learning.

Conclusions

The transition from a proof-of-concept to real-world deployment of GPU-accelerated autonomous vehicle applications necessitates addressing a complex set of engineering and operational challenges. A primary concern is robustness and reliability, as algorithms and hardware must perform flawlessly under a wide range of unpredictable environmental conditions, including extreme temperatures, vibrations, and varying light levels (Gerdes et al., 2018). Furthermore, meeting strict real-time constraints is paramount, particularly for safety-

critical functions like collision avoidance and lane keeping, where even a millisecond of latency can have catastrophic consequences (Papadimitriou et al., 2021). The hardware limitations of GPUs must also be carefully considered, including constraints on memory size, processing power, and thermal management. Onboard vehicle systems require a balance between high performance and low power consumption to prevent overheating and ensure long-term reliability in compact, enclosed spaces (Chen & Wang, 2020).

This project establishes a foundation for several future research directions. One promising avenue involves exploring advanced algorithms, such as deep learning models for semantic segmentation or generative adversarial networks for data augmentation, which can enhance perception capabilities and improve decision-making (Krizhevsky et al., 2017). Integrating with other sensors like LiDAR and radar is crucial for building a more comprehensive and robust understanding of the environment. This multisensor fusion approach can compensate for the limitations of a single sensor, for instance, by allowing the system to see through adverse weather conditions that might obscure a camera's view (Choi et al., 2019). Finally, developing robust safety mechanisms is a non-negotiable step toward deployment. This includes implementing fail-safe procedures, redundancy in processing units, and self-diagnostic capabilities to ensure the system can safely handle hardware failures or software anomalies, thereby protecting occupants and pedestrians (Lee et al., 2020).

Acknowledgments

The authors would like to extend their sincere gratitude to Politeknik Kuala Terengganu and Jabatan Pendidikan Politeknik dan Kolej Komuniti who have made significant contributions to various parts of this research endeavour.

References

Allusse, A., Horain, P., Agarwal, A., & Saipulla, A. (2008). *GpuCV: A GPU-accelerated framework for image processing and computer vision*. Proceedings of the 2008 IEEE International Conference on Multimedia and Expo, 9–12. IEEE. https://doi.org/10.1109/ICME.2008.4607575

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., ... & Zhang, X. (2016). End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316.

Chen, H., & Wang, J. (2020). Thermal management for high-performance computing in autonomous vehicles. IEEE Transactions on Vehicular Technology, 69(8), 8192-8205.

Choi, S., Lee, D., & Kim, J. (2019). A survey on sensor fusion techniques for autonomous vehicles. Journal of Sensors, 2019, 1-15.

Chuttur, Y. ., Kaudeerally, R., & Nazurally, A. (2023). Lane Reconstruction for Self-Driving Vehicles on Dynamic Road Networks. Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 15(2), 29–36. https://doi.org/10.54554/jtec.2023.15.02.004

Dakić, P., & Živković, M. (2021). An overview of the challenges for developing software within the field of autonomous vehicles. In Proceedings of the 7th Conference on the Engineering of Computer Based Systems (ECBS '21) (pp. 1–6). Association for Computing Machinery. https://doi.org/10.1145/3459960.3459972

Duarte, F. (2018). The impact of autonomous vehicles on urban transportation systems. Journal of Urban Technology, 25(4), 3-18.

Fernandes, S., Duseja, D., & Muthalagu, R. (2021). Application of image processing techniques for autonomous cars. Proceedings of Engineering and Technology Innovation, 17, 01-12. https://doi.org/10.46604/peti.2021.6074

Gerdes, J. C., & Thrun, S. (2018). The challenges of real-world deployment for autonomous driving. Science Robotics, 3(24), eaat7144.

Glamočić, D., Lalić, B., & Milanović, D. (2019). *GPU-enabled integral image computation for automotive platforms*

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

Han, T., Li, W., & Zhang, J. (2021). GPU performance optimization for deep learning: A survey. Journal of Parallel and Distributed Computing, 148, 12-25.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539

Lee, J., Kim, H., & Park, S. (2020). A framework for safety assurance of autonomous driving systems. Journal of Advanced Transportation, 2020, 1-12.

Membarth, R., Hannig, F., Teich, J., & Körner, M. (2011). *Frameworks for GPU acceleration: Evaluation of RapidMind, PGI Accelerator, HMPP Workbench, OpenCL, and CUDA*. International Journal of Parallel Programming, 39(5), 417–444. https://doi.org/10.1007/s10766-010-0162-7

NVIDIA. (2023). CUDA: The parallel computing platform and programming model. Retrieved from https://developer.nvidia.com/cuda-toolkit

Padmaraja, V. P., Rohith, R., & Chittesh, S. (2023). Lane detection using image processing for autonomous vehicles. In 2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA) (pp. 1–6). IEEE. https://doi.org/10.1109/ICAECA56562.2023.10200663

Papadimitriou, G., Tsiotras, P., & Vamvoudakis, K. G. (2021). Real-time safety assurance for autonomous systems. IEEE Transactions on Control Systems Technology, 29(5), 2097-2109.

Patterson, D. A., & Hennessy, J. L. (2018). *Computer organization and design: The hardware/software interface*. Morgan Kaufmann.

Pendleton, S. D., Andersen, H., Du, X., Shen, X., Meghjani, M., Eng, Y. H., Rus, D., & Ang, M. H. (2017). *Perception, planning, control, and coordination for autonomous vehicles*. Machines, 5(1), 6. https://doi.org/10.3390/machines5010006

Qualcomm. (2025). *Advanced driver assistance systems: Reducing accidents with AEB and safety features*. Qualcomm Technologies Inc. https://www.qualcomm.com/ \[replace with direct page URL if available]

Saoudi, F. (2022). *Open-source operating systems in autonomous vehicle development: Ubuntu as a case study*.

Shevenell, G. (1984). *Optical and acoustic imaging systems for vehicle control applications*.

Sun, W., & Min, Y. (2023). Research on a driving assistance system for lane changes on foggy highways. Sustainability, 15(13), Article 10032. https://doi.org/10.3390/su151310032

Ubuntu. (2023). The world's most popular free operating system. Retrieved from https://ubuntu.com

Wang, L., Chen, Y., & Liu, Z. (2022). Real-time object detection and tracking with GPU acceleration. IEEE Transactions on Intelligent Vehicles, 7(1), 125-136.

Yang, J. (2018). *Safety-critical real-time GPU principles in NVIDIA-based autonomous systems*.