

Safe Systems Design for Special Purpose Autonomous Vehicles



Thomas H. Drage

BSc *W. Aust.*, BEng (Hons) *W. Aust.*, MIDS *Calif.*

Department of Electrical, Electronic and Computer Engineering
The University of Western Australia

This thesis is presented for the degree of

Doctor of Philosophy

Supervisor: Prof. Dr. rer. nat. habil. Thomas Bräunl

April 2023

Declaration

I hereby certify that:

- This thesis has been substantially accomplished during enrolment in this degree.
- This thesis does not contain material which has been submitted for the award of any other degree or diploma in my name, in any university or other tertiary institution.
- In the future, no part of this thesis will be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of The University of Western Australia.
- This thesis does not contain any material previously published or written by another person, except where due reference has been made in the text and, where relevant, in the Authorship Declaration that follows.
- This thesis does not violate or infringe any copyright, trademark, patent, or other rights whatsoever of any person.

This thesis contains published work and/or work prepared for publication, some of which has been co-authored.



Thomas H. Drage
April 28th 2023

Acknowledgements

I would like to express my sincere thanks to Professor Thomas Bräunl for his support of the projects I have been fortunate to work on as part of the *Robotics and Automation Laboratory* and the *REV Project* as well as the encouragement and guidance I have personally received as I have conducted this research. Thanks must also go to the many excellent undergraduate and master's degree candidates who have collaborated on aspects of the *Formula-SAE Autonomous* project and the *nUWay* shuttle; creating mutual benefit to development of the vehicles and the research goals of all those involved in the projects.

I would like to thank the IEEE, whose excellent societies, particularly the *Intelligent Transportation Systems Society*, have furnished highly useful peer reviews, excellent conferences and quality journals which have been the cornerstone of my own research and of many of the electro-technical advances enjoyed by humanity.

This research is supported by an Australian Government Research Training Program (RTP) Scholarship. Other sponsors of the research deserving of mention include; Galaxy Resources, Xsens, Nvidia, Altronics, SMC, EV Works and SBG Systems.

A deserving special mention must be made to Dr. Kai Li Lim, with whom I collaborated closely on a number of articles and in the co-ordination of the research efforts for which his endless dedication and professionalism was most appreciated. Finally, I would like to thank my partner, Wen, for her encouragement and endurance over the years and my parents whose lifelong support has led me to be writing this today.

Authorship Declaration

In accordance with the regulations of the Graduate Research School, this thesis contains published work and/or work prepared for publications that are co-authored. The bibliographical details of the work and where it appears in the thesis are outlined below.

1. **T. Drage**, J. Kalinowski, and T. Bräunl, "Integration of drive-by-wire with navigation control for a driverless electric race car," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 4, pp. 23–33, 2014.

The estimated percentage contribution of the candidate is 85%.

2. **T. Drage**, T. Churack, and T. Bräunl, "Lidar road edge detection by heuristic evaluation of many linear regressions," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems (ITS)*, 2015, pp. 2465–2470.

The estimated percentage contribution of the candidate is 90%.

3. K. L. Lim, **T. Drage** and T. Bräunl, "Implementation of semantic segmentation for road and lane detection on an autonomous ground vehicle with LIDAR," in *2017 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Daegu, 2017, pp. 429–434.

The estimated percentage contribution of the candidate is 25%.

4. K. L. Lim, **T. Drage**, R. Podolski, G. Meyer-Lee, S. Evans-Thompson, J. Y.-T. Lin, G. Channon, M. Poole, and T. Bräunl, "A Modular Software Framework for Autonomous Vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, Changshu, 2018, pp. 1780–1785.

The estimated percentage contribution of the candidate is 30%.

5. K. L. Lim, **T. Drage**, C. Zhang, C. Brogle, W. W. L. Lai, T. Kelliher, M. Adina-Zada and T. Bräunl, "Evolution of a Reliable and Extensible High-Level Control System for an Autonomous Car," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 3, pp. 396–405, Sept. 2019.

The estimated percentage contribution of the candidate is 30%.

6. **T. Drage**, K. L. Lim, J. E. Hai Koh, D. Gregory, C. Brogle and T. Bräunl, "Integrated Modular Safety System Design for Intelligent Autonomous Vehicles," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, Nagoya, Japan, 2021, pp. 258-265.

The estimated percentage contribution of the candidate is 75%.

7. **T. Drage**, K. Quirke-Brown, L. Haddad, Z. Lai, K. L. Lim and T. Bräunl, "Managing Risk in the Design of Modular Systems for an Autonomous Shuttle Bus," submitted to *IEEE Open Journal of Intelligent Transportation Systems* (under review).

The estimated percentage contribution of the candidate is 65%.

Signed:



Prof. Dr. Thomas Bräunl
*Co-ordinating Supervisor**



Zhihui Lai
Postgraduate Researcher



Dr. Kai Li Lim
Adjunct Research Fellow



Kieran Quirke-Brown
Postgraduate Researcher

*The other co-authors have left UWA and their current contact details are unknown.

Abstract

The dream of fully autonomous, driverless, road vehicles has been envisaged for many years and forms a defining part of any technology driven future. However whilst the development process has accelerated dramatically in the last decade, the driver free commuter road vehicle has not yet been realised. Semi-autonomous (that is, SAE level 1,2 and to some extent level 3) technology is now available, with consumer uptake increasing since around 2020, however it is not without controversy and arguably still under proven. The road towards total driverless automation (SAE level 4+) is paved with many challenges – technical, legal, ethical and commercial and it may well be another decade before the dream is realised.

This research spans a period of significant technological change and has thus evolved with the field, focusing on the development of two “special purpose” autonomous vehicles – vehicles which are not destined for general purpose road use, but instead to achieve full driverless automation within a constrained environment or task set. The *Formula-SAE Autonomous* vehicle, developed from 2013 – 2019 with the goal of fast-laps within an environmentally defined track area underlaid the foundations, which were applied from 2019 onward to the *nUWay* shuttle bus which was designed to operate without a driver in a campus environment, along with pedestrians and other road vehicles. By trading the goal of generalisation for increased automation, the *REV Project* has explored the requirements for highly automated systems and developed a framework which has bought our robotics research from small field-robots to useful full sized vehicles.

The key aspect investigated in this project is system design – with the goals of implementing the required functionality to allow safe and accessible research and development of the automation technology. The application of modular and low-cost systems is explored and evaluated, as well as the key building blocks required to allow safe and reliable navigation. The work examines the use of independent reliable systems to provide safety functionality. Safe systems with advanced sensing are presented as a pathway to driver elimination in our vehicles.

Table of Contents

Declaration	ii
Acknowledgements	iii
Authorship Declaration	iv
Abstract	vi
Table of Contents	vii
List of Acronyms	viii
Introduction	xi
Integration of Drive-by-Wire with Navigation Control for a Driverless Electric Race Car	1-1
LIDAR Road Edge Detection by Heuristic Evaluation of Many Linear Regressions	2-1
Implementation of Semantic Segmentation for Road and Lane Detection on an Autonomous Ground Vehicle with LIDAR	3-1
A Modular Software Framework for Autonomous Vehicles	4-1
Evolution of a Reliable and Extensible High-Level Control System for an Autonomous Car	5-1
Integrated Modular Safety System Design for Intelligent Autonomous Vehicles	6-1
Managing Risk in the Design of Modular Systems for an Autonomous Shuttle Bus	7-1
Conclusion	8-1

List of Acronyms

ABS	Anti-lock braking system
ADAS	Advanced driver assistance system
ADS	Autonomous driving system
AI	Artificial intelligence
AJAX	Asynchronous Javascript and XML
AMCL	Adaptive Monte-Carlo localizer
API	Application programming interface
ARM	Advanced (Acorn) RISC Machine
ASIL	Automotive Safety Integrity Level
AV	Autonomous vehicle
BEV	Bird's eye view
CAD	Computer-aided design
CAN	Controller Area Network
CARLA	Car Learning to Act (simulator)
CCD	Charge-coupled device
CNN	Convolutional neural network
CPU	Central processing unit
CUDA	Compute Unified Device Architecture
DARPA	Defense Advanced Research Projects Agency
DC	Direct current
DDS	Data distribution service
DGPS	Differential GPS
ECU	Electronic control unit
EKF	Extended Kalman filter

EV	Electric vehicle
FAST	Features from accelerated segment test
FMEA	Failure mode and effects analysis
FPS	Frames per second
FS	Functional safety
FSAE	Formula SAE
FTA	Fault tree analysis
GNSS	Global navigation satellite system
GPS	Global Positioning System
GPU	Graphics processing unit
GUI	Graphical user interface
HARA	Hazard and risk assessment
HAZOP	Hazard and operability (study)
HIL	Hardware in the loop
HOG	Histogram of oriented gradients
HTTP	Hypertext Transfer Protocol
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IMU	Inertial measurement system
INS	Inertial navigation system
IO	Input/output
IP	Internet Protocol
IPL	Independent protection layer
ISO	International Organization for Standardization
LIDAR	Light detection and ranging
LOPA	Layer of protection analysis
MCC	Matthews correlation coefficient
MEMS	Micro electromechanical system
ML	Machine learning
MTBF	Mean time before failure

NMEA	National Marine Electronics Association
OEM	Original equipment manufacturer
PAS	Publicly available specification
PC	Personal computer
PHA	Preliminary hazard analysis
PID	Proportional-integral-derivative (controller)
PLC	Programmable logic controller
Protobuf	Protocol buffer
PWM	Pulse width modulation
RAM	Random access memory
REV	Renewable energy vehicle
RGB	Red-green-blue
ROS	Robot Operating System
RTK	Real-time kinematic
RTOS	Real-time operating system
SAE	Society of Automotive Engineers
SBAS	Satellite based augmentation system
SLAM	Simultaneous localisation and mapping
SOTAI	Safety of the AI
SOTIF	Safety of the intended functionality
STPA	System theoretical process analysis
SVM	Support vector machine
TCP	Transport Control Protocol
TEB	Time elastic band
TI	Texas Instruments
UI	User interface
USB	Universal serial bus
UWA	The University of Western Australia

Introduction

Autonomous and Intelligent Vehicles

Increasing levels of different varieties of vehicular automation have been emerging for several decades; some types of which are highly refined and ubiquitous in their niches (e.g. driverless trains [1], warehouse robots [2]) although the automation of these types is usually characterised by a lack of intelligence and reliant upon fixed tracks, sensors and programmes. Meanwhile, despite the long predicted decline of the profession, humans are heavily employed as drivers of vehicles of general purpose nature which operate in semi-controlled surroundings; as truck drivers, locomotive engineers, couriers and other roles which underpin the supply chains of all societies and the operation of every other kind of commercial endeavour [3]. In our private lives, we utilise our intelligence to execute our commute, visit friends and transport supplies to our homes. Eliminating this practice is highly attractive – it will reduce costs for businesses, give back time to consumers and potentially improve safety, contributing to a worthwhile improvement in quality of life across the globe [4].

Thus, the opportunity for automation in the land transport market is significant – intelligent autonomous vehicles will transform the concept of mobility, particularly for the multitude of consumers who own and drive cars as part of their daily lives and change the way they procure and value transportation options. It is thought that the availability of automation will cause a shift towards uptake of shared mobility rather than individual car ownership [5] and that this will drive improved energy and emissions outcomes [6] but at the same time increase urban sprawl [7] while improving mobility overall [8]. As well as increased adoption of ride or car sharing schemes, it is expected that autonomous last-mile services will continue to grow in popularity for both transport of people [9] and delivery of goods [10] with benefits to traffic congestion, efficiency and convenience. The commercial adoption of this technology is of extremely high value to the automotive industry; interested consumers in some segments are willing to pay a

large premium to obtain autonomous driving functions. Software functionality and feature upgrades will become critical to marketing a vehicle and will open the door to after-market subscription and upgrade revenues. The capabilities required to deliver high level automation will require a change in the nature of the industry, but could be worth as much as \$400 billion in the passenger car market alone by 2035 [11].

The *REV Project*, in collaboration with the *Robotics and Automation Laboratory* at The University of Western Australia is focussed on advancing electromobility and creating intelligent transportation systems in order to improve the environmental and public safety outcomes associated with road transportation [12]. Two “special purpose” electric autonomous vehicles were developed as part of the work which forms this dissertation. The vehicles are special purpose in the sense that they are not trucks or passenger vehicles for use on public roadways, however, the *REV Project’s* goal is to utilise these vehicles as research platforms to develop techniques which are applicable to the problem of mass road transportation. The key to this is the goal of developing intelligent vehicles which utilise advanced sensing, perception and localisation methods to allow the vehicle to operate in a changing environment and to respond dynamically while achieving a high level goal (e.g. arrival at a new destination) as opposed to merely achieving automation.

One significantly desirable aspect of automation of the road transport system is the improvement in safety by means of eliminating the opportunity for human error which results from a variety of factors including fatigue, intoxication, distraction and skill [13]. The basis for the presumption that an autonomous car is safer is consistency of performance and unwavering compliance with rules. However, in reality the dynamically changing nature of public roads, for example by unexpected ingress of a pedestrian to the roadway or the bursting of a tyre on another vehicle, means it is not possible to avoid all incidents and making an automated decision which minimises harm is legally, ethically and practically complex [14]. At a more basic level the automation itself must be sufficiently reliable that it doesn’t make an error under reasonably expected conditions or fail to operate during navigation in a way which in fact causes an accident or near miss; data collected from various autonomous driving trials has shown that this has not yet been achieved and an autonomous “driver” is many times more dangerous than the average human [15].

Mass produced passenger cars have incorporated levels of automation since the introduction of adaptive cruise control in the 1990s and advances in cost of sensors and computational abilities have driven widespread adoption of advanced driver assistance systems (ADAS) in the last decade. Indeed, such features have tended towards being labelled as autonomous driving however they all require the presence of an alert supervising human driver and are incapable of executing a high level journey goal [16]. The Society of Automotive Engineers (SAE) defines “Levels of Driving Automation”, from zero to five, with the lower three being considered driver support and the upper automated driving [17]. The Autopilot and Full Self-Driving offerings from American car manufacturer Tesla achieve just SAE Level 2, whilst Level 4/5 systems, which will lead the way to making human drivers redundant may become publicly available to the market in the late 2020s [18]. The challenge of achieving fully automated intelligent driving is still significant, with large car manufacturers grappling with the complexity and cost [19] – thus, the scope for research and development of components of such a solution is as still as significant as it was at the beginning of the *REV Project*’s efforts and the special purpose vehicles continue to offer an accessible avenue for this work.

Emerging Standardisation

The introduction of commercial SAE Level 3+ vehicles to the public roadway requires the implementation of robust regulatory frameworks in order to ensure that they result in improved safety outcomes. These regulations will rely on the implementation of quality technical standards, particularly with respect to safety [20] features of the vehicles and their automation systems. Such standards have been published over the last decade and continue to emerge and be updated as the state of the art progresses and commercial readiness emerges. International regulations have also been published recently, including UN-R157 [21] on the basis of which Mercedes-Benz has been granted approval for their now commercially available SAE Level 3 DRIVE PILOT system [22]. Mercedes-Benz publicise their application of standards [23] including ISO 26262 [24] and ISO/PAS 21448 [25] to facilitate the safety assessment of their self driving platform, as do other car manufacturers such as Ford [26] who seek to develop autonomous driving systems (ADS) which are able to be validated and approved when their products are ready for release to the consumer market. Thus, standardisation and regulatory bodies

and car manufacturers have a vested interest in the development and ratification of such technical standards.

Amongst the oldest of the standards specifically dealing with safety of electronic systems in vehicles is ISO 26262 which was first published in 2011 and describes the concept of functional safety of electrical and electronic systems in road vehicles. It is derived from the functional safety standard IEC 61508 [27] and introduces a safety life cycle for control of systematic failures of electrical /electronic/programmable devices and the probabilistic assessment of risks associated with random hardware failures. In 2019, this was augmented by ISO/PAS 21448 which introduces the concept of “safety of the intended functionality” and was originally targeted at SAE Level 1 and 2 systems but updated in 2022 to target all levels of driving automation. In this project we describe the implementation of safety functionality as part of the ADS for two vehicles and examine the applicability of these standards to our special purpose vehicles as well as exploring the extensions required to deal with fully autonomous intelligent driving.

Formula-SAE Autonomous Project

The University of Western Australia has long participated in the Formula-SAE student motor-sport competition, building a number of highly successful internal combustion race cars [28]. The *REV Project* then sought to develop prototype electric-drive Formula-SAE type cars as part of ongoing research into electro-mobility. Two such cars were developed prior to the introduction of an official Formula-SAE electric racing class, one of which was selected for conversion to drive-by-wire [29], once again, before an autonomous driving class existed within the SAE’s student racing programme.

The *Formula-SAE Autonomous* vehicle features a 48V 13 kW electric drive system capable of propelling the car to 70 km/h, with a 4.3 kWh lithium-iron-phosphate battery. Drive-by-wire was implemented with control of the electronic throttle via an embedded system, a servo-operated brake and electric motor driven steering controller. An array of sensors initially comprising a four-layer automotive LIDAR, Inertial Measurement Unit (IMU) and GPS receiver were installed and augmented during this project by binocular computer vision system, four-wheel odometry, upgraded INS/GNSS system and two additional LIDAR sensors. Whilst

relatively compact and light-weight, the vehicle dynamics are significantly closer to a passenger car than platforms traditionally used for field robotics research. This, combined with the comprehensive sensor system results in a highly capable research platform for autonomous driving.

The original architecture for the *Formula-SAE Autonomous* vehicle comprised a laptop computer, embedded drive-by-wire controller, embedded safety monitoring controller and hard-wired interlocks. Through this project, the system evolved to use multiple high performance embedded computer platforms, driven by advances in low-power processors, culminating in the implementation of the NVIDIA Jetson TX1 and Xavier AGX platform which enabled computer vision applications. The safety and human interface systems were subject to continuous improvement and all re-implemented to achieve better reliability and diagnostic capability. The final iteration of the architecture for the *Formula-SAE Autonomous* vehicle's control and safety systems was carried over to the *nUWay* shuttle bus project for further development.

The *Formula-SAE Autonomous* platform was utilised to develop and test both LIDAR and computer-vision based navigation systems, way-point based navigation and dynamic path-planning algorithms, and perception algorithms utilising machine learning. The navigation control software evolved from a monolithic multi-threaded C++ application, to an innovative modular C++ application to a hybrid utilising the capabilities of the Robot Operating System (ROS) as its improved capabilities and the availability of sufficient on-board computational power made such an architecture practical and beneficial. Whilst the race car platform was highly useful as a research platform, there is limited application of a such a vehicle to society and hence the *REV Project* sought a more advanced vehicle with the goal of applying the systems developed to solving a human electro-mobility problem.

nUW_Ay Shuttle Bus Project

Autonomous shuttle buses are a common early commercial incarnation of an autonomous vehicle and are available from manufacturers such as Navya and EasyMile. They generally feature six to eight seats and operate at relatively low speed as a novelty or technology demonstration, particularly at tourist sites [30]. Such vehicles operate by means of navigating between pre-defined waypoints using a Differential GPS (DGPS) or Real-time Kinematic GPS (RTK-GPS) unit for localisation augmented by LIDAR or computer vision systems which provide feedback via a communications link and may trigger actions such as slowing or stopping if a hazard is encountered [31]. The vehicles are typically staffed with a safety officer or monitored by a remote operations centre. The *REV Project* identified an application for such a shuttle in moving students from distant locations around the campus whilst creating a platform for research into autonomous driving and electro-mobility.

The *nUW_Ay* shuttle is based on an EasyMile EZ10 autonomous electric shuttle bus [32] which was purchased without the software components usually offered as a service by the manufacturer. Thus, the EZ10 provided the mechanical and electrical infrastructure for clean-slate implementation of an intelligent navigation control system, with the goals of full autonomy within the campus environment; sufficient to work without a prescribed way-point route and the ability to navigate around obstacles such as pedestrians, parked vehicles, temporary events and other common campus features. Furthermore, the system should offer the opportunity to operate in a safe and reliable fashion independent of off-board processing, a monitoring centre or on-board safety officer; with telemetry only used for calling the shuttle and gathering operational statistics.

As supplied, the EZ10 offers a 48V 4kW reversible electric drive system, with redundant and fail-safe braking actuators, and four wheel steering. It can achieve a maximum speed of 40 km/h and has a turning circle with radius less than 4 m [33]. It has 8 kWh of lithium-iron-phosphate cells in the main battery and the charging system was modified to be compatible with the *REV Project's* charging station network. The shuttle has mounting positions for four single line safety LIDARs at the corners, two 4 layer forward and backward facing front and back as well as two 16 layer LIDARs providing 30° vertical and 140° horizontal field of view. In

addition, two monocular, monochrome cameras were provided and an Inertial Navigation System (INS) with built in sensor fusion by means of a dual antenna Global Navigation Satellite System (GNSS) receiver was added to the vehicle. This results in an extremely useful set of instrumentation for implementation of redundant perception and localisation capabilities for reliable and safe navigation by the vehicle.

The shuttle bus was supplied with no control software and the drive system utilised a proprietary set of instructions via a Controller Area Network (CAN) bus. However, an interface for manual control was provided and this was utilised in this project to interface control and safety functions via an embedded system, with navigation control provided by a ROS based architecture ported from the *Formula-SAE Autonomous* vehicle. An NVIDIA Jetson AGX Xavier embedded computer was utilised as an accelerated compute node to allow the processing of computer vision and LIDAR sensor inputs in real time for navigational purposes; something which the shuttle's original architecture was not capable of.

The architecture developed in this project was integrated with the shuttle bus and utilised to provide a safeguarded means to develop the navigation control system within the ROS framework [34]. A hazard and risk assessment (HARA) methodology was devised along with a workflow for its use during development of the navigation system; workshops were conducted with the purpose of identifying the safety requirements to achieve full (i.e. SAE level 5) autonomous of the *nUWay* shuttle. The proposed safeguards comprise low-level hardware features as well as advanced algorithms, combined in such a way that reduces the likelihood of a hazard leading to an unsafe consequence. The ability to achieve the implementation of such safeguards is dependent upon the architecture of the system and the application of a quality system, and will allow the *nUWay* shuttle to transition from requiring an alert safety driver to fully driverless operation in the future.

Contributions

The works comprising this dissertation describe the creation of the necessary components and methods for the development of two special purpose autonomous vehicles. The primary purpose is investigation of requirements and solutions for system architectures which allow the goals of the vehicle to be achieved whilst maintaining the ability to be safe and reliable during development of components as well as in their final assemblage. The primary contributions are summarised as follows:

- The development of an autonomous Formula-SAE car, consisting of a control system utilising LIDAR, inertial and GPS sensors for navigation operating via a drive-by-wire controller and in tandem with an independent safety system. [Chapter 1]
- The implementation of a real-time road edge detection system, intended to be use for ensuring a vehicle is safely constrained to a roadway, based on LIDAR technology and statistical modelling. [Chapter 2]
- A comparison of a computer-vision method of object detection for navigation with the capabilities of LIDAR, with a view to complementary application of the two technologies for navigation. [Chapter 3]
- An improved software framework which allows greater flexibility for modular development of navigation and control components, whilst maintaining performance and the integrity of the safety monitoring functions. [Chapter 4]
- The migration to and expansion of the framework within the Robot Operating System (ROS), including the application of machine learning methods and visual navigation within the system, forming a sophisticated and capable system. [Chapter 5]
- A review of methods for design of safe vehicle control systems, a novel process for the assessment of the safety requirements for a special purpose autonomous vehicle and implementation of a system design on two vehicles which allows these requirements to be achieved. [Chapter 6]
- A review of the outcome of a detailed safety assessment using the previously described method with a view to achieving full autonomy of a shuttle bus

and description of the components implemented to ensure the safe function of the vehicle. [Chapter 7]

Chapter Synopses

This dissertation comprises seven articles, the first five relate to the development of the *Formula-SAE Autonomous* vehicle and detail the system design and components within it. The latter two articles are centred on the creation of a framework for the *nUWAg* shuttle bus and the development and application of a methodology for creation of a system design which can implement the safety requirements for this more complex, fully autonomous, vehicle. The systems described evolved over the course of the project; the evaluation of components which were investigated is located in the respective chapters and a description of the evolution and findings is presented in the conclusion.

Chapter 1 presents a review of historical vehicle automation and the world-first application of drive-by-wire with a navigation control system to a Formula SAE race car. The resulting system is presented as a platform for research into autonomous driving, which can be easily and cost-effectively replicated. Actuators, sensors and an embedded system are described in addition to a sophisticated, bespoke control software which is able to run in real-time on-board the vehicle. Safety interlocks and active safety systems are presented which ensure the safe operation of the vehicle with or without a driver.

Chapter 2 presents a novel algorithm for road edge detection which is performant on both well defined and informally defined roadways utilising LIDAR sensing on a moving vehicle. The algorithm incorporates inertial sensor data for improved stability and is tested in several scenarios.

Chapter 3 explores the additional capabilities offered by computer vision to a road-based autonomous vehicle in providing information on visual road markings and classification of objects in the vehicle's trajectory. The results are contrasted with LIDAR measurements and show a good fit for vision augmentation of a LIDAR-driven vehicle.

Chapter 4 presents a unified modular software framework which supersedes the original navigation system of the *Formula-SAE Autonomous* vehicle with a view to allowing improved ease of expansion of the system's functionality and the

ability to perform more rigorous off-line software testing while maintaining the safety mechanisms of the previous monolithic system.

Chapter 5 details the implementation of the navigation system for the *Formula-SAE Autonomous* vehicle utilising a hybrid system, consisting of the previously presented software framework and a node-based system utilising ROS. This approach allows the use of open-source and community supported packages resulting in a more rapid development process in terms of overall functionality of the vehicle. It enables the integration of an off-line hardware-in-the-loop simulation system and nodes implementing machine learning algorithms for perception and navigation.

Chapter 6 presents a review of the standards and methods available for assessment and requirements definition of safety functions for autonomous vehicles. A new methodology is presented which is able to be integrated into the development work conducted on experimental autonomous vehicles. The system design for safety and control of two operational autonomous vehicles is presented and a multi-level solution for safe operation in a complex environment.

Chapter 7 documents the results of application of the previously developed methodology with respect to the *nUWay* shuttle bus with the goal of achieving full driverless autonomy in a safe and reliable fashion. The benefits and shortcomings of a ROS based system are evaluated. Results of initial testing of the system are presented and show an effective overall system design can augment the issues associated with an extensible modular control system in a fully autonomous vehicle.

References

- [1] Y. Wang, M. Zhang, J. Ma, and X. Zhou, ‘Survey on Driverless Train Operation for Urban Rail Transit Systems’, *Urban Rail Transit*, vol. 2, no. 3, pp. 106–113, Dec. 2016.
- [2] P. R. Wurman, R. D’Andrea and M. Mountz, "Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses," *AI Magazine*, vol. 29, (1), pp. 9-19, 2008.
- [3] M. Gittleman and K. Monaco, “Automation Isn’t About to Make Truckers Obsolete”, *Harvard Business Review*, Sept. 2019. [On-line]. Available: <https://hbr.org/2019/09/automation-isnt-about-to-make-truckers-obsolete>
- [4] Y. Yin, “Perspective Chapter: Future Perspectives of Intelligent Autonomous Vehicles” in *The Dynamics of Vehicles - Basics, Simulation and Autonomous Systems*. IntechOpen, Oct. 2022.
- [5] N. Menon et. al., “Shared autonomous vehicles and their potential impacts on household vehicle ownership: An exploratory empirical assessment,” *International Journal of Sustainable Transportation*, vol. 13, no. 2, pp. 111–122, 2019.
- [6] K. Akimoto, F. Sano and J Oda, “Impacts of ride and car-sharing associated with fully autonomous cars on global energy consumptions and carbon dioxide emissions”, *Technological Forecasting & Social Change*, vol. 174, Jan. 2022.
- [7] J. Guan et. al., “Potential Impacts of Autonomous Vehicles on Urban Sprawl: A Comparison of Chinese and US Car-Oriented Adults”, *Sustainability*, vol. 13, no. 14: 7632, 2021.
- [8] L. T. Truong et al, "Estimating the trip generation impacts of autonomous vehicles on car travel in Victoria, Australia," *Transportation*, vol. 44, no. 6, pp. 1279-1292, 2017.
- [9] R Abe, “Preferences of urban rail users for first- and last-mile autonomous vehicles: Price and service elasticities of demand in a multimodal environment”, *Transportation Research Part C: Emerging Technologies*, vol. 126, May 2021.
- [10] M. Figliozzi and D. Jennings, “Autonomous delivery robots and their potential impacts on urban freight energy consumption and emissions”, *Transportation Research Procedia*, vol. 46, pp. 21-28, 2020.
- [11] J. Deichmann et. al., “Autonomous driving’s future: Convenient and connected”, McKinsey & Company, Jan. 2023. [On-line]. Available: <https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/autonomous-drivings-future-convenient-and-connected#/>
- [12] T. Bräunl. “The REV Project.” The REV Project. [On-line]. <https://therevproject.com/> (accessed Mar. 26 2023)
- [13] J. M. Anderson et. al., *Autonomous Vehicle Technology: A Guide for Policymakers*. Santa Monica: RAND Corporation, 2014, ch. 2.
- [14] J. Fleetwood, “Public health, ethics, and autonomous vehicles,” *American Journal of Public Health*, vol. 107, no. 4, pp. 532–537, 2017.
- [15] S. S. Banerjee et. al., "Hands Off the Wheel in Autonomous Vehicles?: A Systems Perspective on over a Million Miles of Field Data," *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Luxembourg, 2018, pp. 586-597.
- [16] M. Murtaza, C.-T. Cheng, M. Fard, and J. Zeleznikow, “The importance of transparency in

naming conventions, designs, and operations of safety features: from modern ADAS to fully autonomous driving functions,” *AI & society*, 2022.

- [17] “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles”, SAE J3016_202104, Society of Automotive Engineers, Warrendale, USA, Apr. 2021.
- [18] T. Litman, “Autonomous Vehicle Implementation Predictions,” Victoria Transport Policy Institute, Victoria, British Columbia, Canada, Jan. 23 2023. [Online]. Available: <https://www.vtpi.org/avip.pdf>
- [19] H. Guinness, “A major player in the AV space is hitting the brakes—here’s why”, *Popular Science*, Oct. 2022. [On-line]. Available: <https://www.popsoci.com/technology/argo-ai-shut-down/>
- [20] A. M. Ivanov, S. S. Shadrin and D. A. Makarova, "The Analysis of International Standards in the Field of Safety Regulation of Highly Automated and Autonomous Vehicles," *2022 Systems of Signals Generating and Processing in the Field of on Board Communications*, Moscow, Russian Federation, 2022, pp. 1-6.
- [21] UN Regulation No. 157, “Uniform provisions concerning the approval of vehicles with regard to Automated Lane Keeping Systems,” United Nations Economic Commission for Europe, Geneva, CH, Standard, March 2023.
- [22] M. Minielly, A. Berg and C. Decker, "Mercedes-Benz world’s first automotive company to certify SAE Level 3 system for U.S. market," *Business Wire*, Jan. 2023. Available: <https://www.proquest.com/wire-feeds/mercedes-benz-world-s-first-automotive-company/docview/2769612413/se-2>.
- [23] “Introducing DRIVE PILOT: An Automated Driving System for the Highway,” Mercedes-Benz, Mar. 2023. [On-line]. Available: <https://group.mercedes-benz.com/dokumente/innovation/sonstiges/2023-03-06-vssa-mercedes-benz-drive-pilot.pdf>.
- [24] “Road vehicles — Functional safety,” International Organization for Standardization, Geneva, CH, Standard, Aug. 2018.
- [25] “Road vehicles — Safety of the intended functionality,” International Organization for Standardization, Geneva, CH, Standard, Jan. 2019.
- [26] “A matter of trust 2.0 – Ford’s approach to developing self-driving vehicles”, Ford Motor Company, Jun. 2021. [On-line]. Available: <https://media.ford.com/content/dam/fordmedia/North%20America/US/2021/06/17/ford-safety-report.pdf>
- [27] “Functional safety of electrical/electronic/programmable electronic safety-related systems,” IEC, Geneva, CH, Standard, Apr. 2018.
- [28] “UWA Motorsport.” The University of Western Australia. [On-line]. Available: <https://www.uwa.edu.au/Facilities/Motorsport> (accessed Mar. 26 2023)
- [29] J. Kalinowski, T. Drage, and T. Bräunl, ‘Drive-By-Wire for an Autonomous Formula SAE Car’, IFAC Proceedings Volumes, vol. 47, no. 3, pp. 8457–8462, 2014.

- [30] “Lessons learned from automated vehicle trials in Australia,” National Transport Commission, Melbourne, Victoria, Australia, Dec. 2020. [On-line]. Available: <https://www.ntc.gov.au/sites/default/files/assets/files/AV-lessons-learned-2020.pdf>
- [31] C. Iclodean, N. Cordos, and B. O. Varga, “Autonomous Shuttle Bus for Public Transportation: A Review,” *Energies*, vol. 13, no. 11, p. 2917, Jun. 2020.
- [32] “EZ10 passenger shuttle.” EasyMile. [On-line]. Available: <https://easymile.com/vehicle-solutions/ez10-passenger-shuttle> (accessed Mar. 26 2023)
- [33] EasyMile, “EasyMile EZ10 v2 Cybercar User Manual,” EasyMile, User Manual Rev B04., Iter. 105, Dec. 2015.
- [34] M. Quigley et. al., “Ros: an open-source robot operating system,” *ICRA workshop on open source software*, vol. 3, no. 3.2, Kobe, Japan, 2009, p. 5.

CHAPTER 1

Integration of Drive-by-Wire with Navigation Control for a Driverless Electric Race
Car

Integration of Drive-by-Wire with Navigation Control for a Driverless Electric Race Car

**Thomas Drage, Jordan Kalinowski,
and Thomas Bräunl**

*Renewable Energy Vehicle Project (REV),
School of Electrical, Electronic and Computer
Engineering, The University of Western Australia,
Perth. E-mail: Thomas.Braunl@UWA.edu.au*

Abstract—This article presents the design and implementation of a drive-by-wire system and a navigation control system for an autonomous Formula SAE race car. The result is the development of a platform for research into autonomous driving which can be easily replicated. Drive-by-wire actuators for acceleration, braking and steering of the vehicle are discussed, as well as the embedded low-level control system. The high-level navigation system features sensor fusion of a 6-dof IMU with a standard GPS and the integration of an automotive LIDAR. Operation of the vehicle is via a multi-threaded program with asynchronous IO and is based upon recording and driving waypoints. In addition to independent safety interlocks, active safety systems are an integral part to both the drive-by-wire and navigation systems.

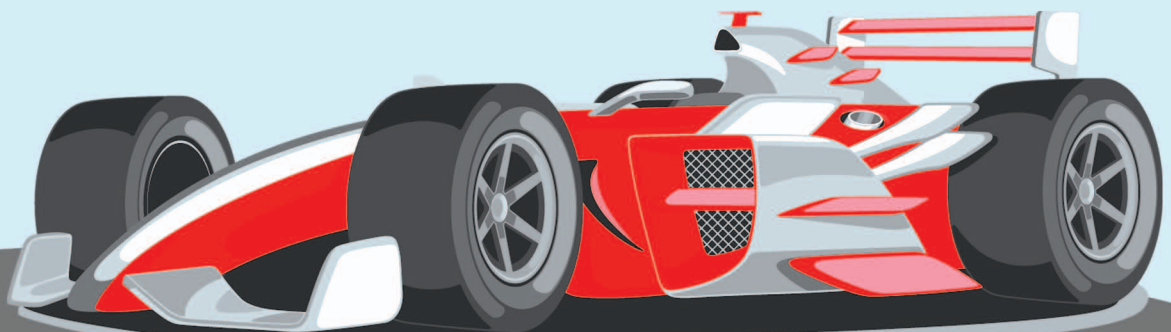


IMAGE LICENSED BY GRAPHIC STOCK

Digital Object Identifier 10.1109/MITS.2014.2327160
Date of publication: 27 October 2014

I. Introduction

Formula SAE is a long-running annual competition organized by the Society of Automotive Engineers with competitions events in the U.S., Europe, and Australia.

In former years Formula SAE (FSAE) has been a design competition only for petrol cars, but recently the new class FSAE-Electric has been introduced [1].

In addition to two road-registered electric vehicles (EVs), a Hyundai Getz and a Lotus Elise, UWA's Renewable Energy Vehicle Project (REV) has built two electric Formula SAE cars. The one discussed here features two electric motors driving each of the two rear wheels via independent controllers and has full drive-by-wire control of the throttle, steering and the hydraulic braking system. This project is outside the scope of the FSAE competition, which neither allows drive-by-wire nor autonomous drive systems, but it is an entry-level platform for research into driverless cars. The use of a Formula-SAE car provides several advantages for such a project; the car is mechanically simple, Formula-SAE cars with similar designs are common at universities worldwide and the size of the car makes testing accessible without forgoing race car vehicle dynamics. Furthermore, the use of an electric vehicle makes the project significantly more practical for student-work as the risks and environmental issues associated with petrol-engine cars are eliminated and the systems installed in this project are able to take advantage of the large amount of electrical energy available on the vehicle already. The car builds on our previous research on driver-assistance systems brake-by-wire and steer-by-wire for a BMW X5 [2].

For the driverless FSAE project, the goal was to be able to autonomously drive a vehicle around a race track. This is being achieved by placing waypoints along the ideal driving line, as well as "fence points" to lock out non-driving areas. Maps can either be recorded by driving the course manually or through a Google Maps driven web-interface. During autonomous driving a laser scanner is used for detection of road edges as well as any obstacles on the track. Safety sys-

tems are essential for a driverless system, as the car weighs in excess of 250 kg and is capable of driving at a speed of 80 km/h. Both the low-level drive-by-wire, as well as high-level navigation system have independent safety systems built in. These include remote intervention, remote heart-beat and remote emergency stopping, which are implemented through a wireless link to a base station as well as through wired control buttons on the car itself.

II. Autonomous Automobiles and Driver-Assistance Functions

As manufacturers in the automotive industry continue to take advantage of new technologies the various functions of a vehicle tend to become increasingly automated [3]. While various designs have been considered for such systems the basic principle behind them is the same: when a driver attempts to control the vehicle, feedback from sensors detect and provide data to control system based on the driver's input, or the judgments of driver assistance systems.

Hydraulic systems are typically used to assist a driver to steer a vehicle, with electric assistance becoming more popular in order to take advantage of characteristics such as lower power consumption [4]. Electronic control allows development of algorithms that provide capabilities such as keeping a vehicle inside a lane when there is no driver input to increase road safety [5]. While a full steer-by-wire system would reduce driver steering effort considerably legal concerns prevent a full steer-by-wire implementation [5], [6]. In the event of a failure, it is preferable that a mechanical link is still present, so that control of steering is still possible.

Commercial applications of brake-by-wire systems have allowed reduced effort in applying braking forces. Brake control system use feedback from multiple sensors such as hall-effect sensors that convert brake pedal displacement into an electronic signal [7]. Electronic control systems are used to improve braking performance as factors such as current wheel speed can be measured from sensors and various control methods (such as new control algorithms, ABS and electronic stability programs) can be developed [8] [9]. Resistance on the brake pedal is still felt by the driver and failure of the brake control unit deactivates electronic control systems while still allowing manual hydraulic forces to be applied to the brakes [8].

Throttle-by-wire systems use a similar method to that of brake-by-wire systems in that a sensor that outputs a voltage proportional to the amount that the accelerator pedal is depressed is used as feedback for throttle control. The accelerator pedal is not mechanically linked to throttle control as in earlier mechanical designs [10]. Instead, an electric motor acts upon a throttle valve to allow alteration of acceleration rate. Electronic control of engine throttle allows more efficient operation of the motor through the use of electronic control systems [10]. Furthermore, sensor failure does not



FIG 1 UWA/REV formula SAE-electric car with drive-by-wire.

result in a runaway vehicle: upon detection of throttle sensor failure, the output from the vehicle's motor is limited via a mechanical mechanism [11].

Over the last decade driver assistance systems have gradually become standard in new cars though most current offerings are of limited sophistication. Adaptive cruise control utilizing a laser sensor was first offered by Toyota in 1998, with systems designed to preempt potential crashes becoming available on Mercedes-Benz models in 2002 [12]. Since then more advanced camera-based systems such as lane-keeping assistance systems and driver drowsiness detection have become commonplace [12]. At UWA, a driver assistance system with drive-by-wire control of the steering and brake was implemented using optical and LIDAR sensing on a BMW X5 [2]. This, like most assistance systems, is minimally invasive and designed simply to augment the shortcomings of the human driver and does not feature any autonomy. The future of this technology holds significant promise for improving road safety and is exemplified by research at Daimler, which offers novel functionality such as the detection of dangerous situations in roundabouts [13].

Research into autonomous vehicles began in the 1980s with projects such as the EUREKA Prometheus Project in Europe and the United States' Autonomous Land Vehicle Project [14]. The DARPA Grand Challenges [15], [16] in 2004 and 2005 saw teams of autonomous vehicles competing to navigate a desert environment whilst the 2007 Urban Challenge [17]–[19] required navigation of a road based course and adherence to traffic protocols. In Europe, the VisLab Intercontinental Autonomous Challenge in 2010 [20] required an autonomous drive following a leader car from Italy to China. These competitions saw massive development of the field, with advanced technologies already becoming available for automotive use. Autonomous driving technology is evolving rapidly and is well on its way to finding commercial use in years to come. Google recently revealed that their fleet of autonomous cars had travelled 140,000 miles on US public roads without human intervention [21]. In Australia, Rio Tinto plans to have 150 autonomous trucks supplied by Komatsu working in their Pilbara mining operations by 2015 [22].

Recently technologies have matured and research into the potential of autonomous cars in racing has begun, with projects such as Stanford University's autonomous Audi TTS, which has been able to perform as well as seasoned racing drivers [23]. This project is of particular interest as its aims in using electronic control systems to drive "at the limits" of the car's mechanical abilities are similar to our project. A sophisticated suite of navigation sensors are used and have seen the car drive complex, long (20 km) race courses [24]. It is notable however, that many of the pioneering projects discussed here feature traditional, petrol power vehicles and have not taken advantages offered by simple electric vehicles such as the Formula-SAE Electric class.

III. Drive-by-Wire Systems

A. Brake-by-Wire

Brake-by-wire systems are prohibited on traditional FSAE vehicles, as are electronic systems that assist braking (as commonly found in commercial vehicles). This complicated the introduction of a brake-by-wire system as existing electronic systems could not be manipulated to create a brake-by-wire system, instead, a servomotor was installed. The brake pedal on the FSAE vehicle drives a hydraulic reservoir which applies a force to the front disc brakes. In order to convert the rotary movement of the servomotor into movement appropriate to pull back the brake pedal, brackets were designed to turn the servomotor's rotary movement into an arc, which led to the design of a series of brackets (see Figure 2). The bracket which is mounted on the rear of the brake pedal has a slot for coupling to the rest of the linkage. This slot allows the driver/passenger to apply the brake further than the autonomous system has engaged them if necessary. It also decouples the linkage from manually driving the servomotor when the vehicle is being driven manually and the driver actuates the brake.

B. Steer-by-Wire

No existing steer-by-wire systems could be taken advantage of, as traditional FSAE rules prohibit the use of steer-by-wire

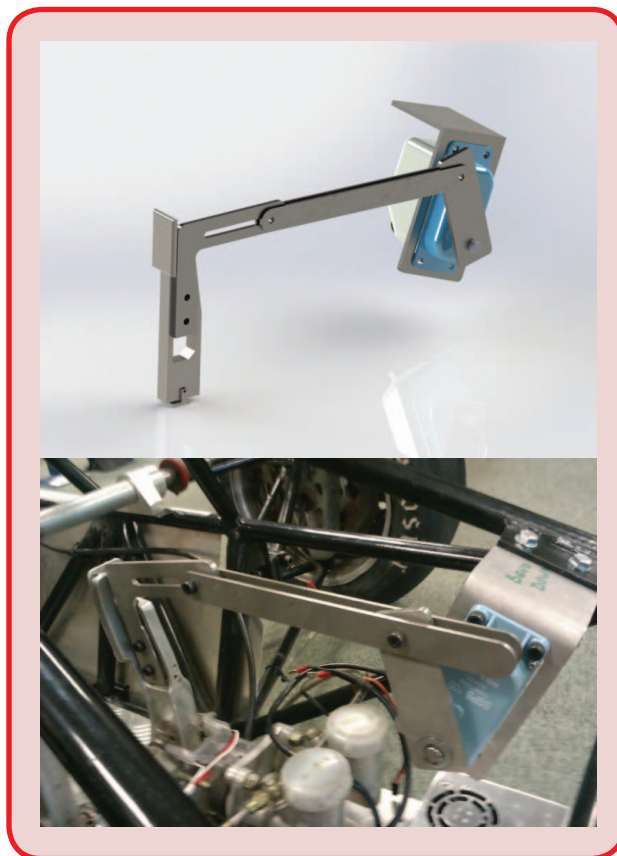


FIG 2 CAD design and assembled brake actuator system.

Using a DC motor and avoiding the use of a worm gear driven system allows the driver of the vehicle to steer in emergency situations while the autonomous system is functioning.

and having a long lifespan as it does not rely on a mechanical wiper were key factors in its selection. Gears attached to the steering column and the sensor allow it to read the absolute steering position at all times. Figure 3 shows the hall sensor attached to the steering column by gear drive. Sensor feedback is used to control the DC motor via a PID loop.

technology. Any steer-by-wire systems installed will have to be capable of acting on the vehicle with similar effort to that of a human. To perform actuation of the steering function of the car, a 12 V electric motor with an integrated gearbox was installed. As the maximum torque required to steer a vehicle is at stationary [25], the combination used met requirements to steer the vehicle when not in motion. Using a DC motor and avoiding the use of a worm gear driven system allows the driver of the vehicle to steer in emergency situations while the autonomous system is functioning. This setup also allows the vehicle to be driven normally without decoupling the DC motor from the steering column. While the addition of the steering actuator does introduce the need for extra steering effort when a driver attempts to steer, the vehicle can be returned to its original steering effort (such as for driver training) by decoupling the belt.

A mild steel plate was designed to mount the motor and gearbox combination to the vehicle. To ensure that the car could still be driven manually, this motor was mounted above the driver's legroom next to the steering column. To couple the motor's output to the vehicle's steering system, a belt drive was used with pulleys on either end. Shown in Figure 3, the motor mounting bracket also provides belt-tensioning capabilities as the bracket itself is slotted. Care had to be taken to ensure that any systems installed did not compromise the safety of a passenger in the SAE vehicle as no protective barriers exist between the passenger and installed systems and because little space is allowed for a person due to the nature of the vehicle.

For steering-angle feedback a Honeywell HRS100SSAB-090 rotary hall sensor was used. Being cost-effective, accurate

C. Accelerate-by-Wire

All autonomous drive systems installed in our FSAE car are powered from a 48 V-to-12 V DC-DC converter which isolates the system from the existing battery pack and drive motors and provides a generous 350 W for use by the autonomous drive systems. The existence of an electronic throttle in this vehicle was advantageous; an analogue output signal between 0 V and 5 V was required to replicate the accelerator pedal hall sensor output and is easily switched between automatic and manual control. A further advantage of such a system is that the signal scaling can be adjusted in software to provide a variable power range which was utilized to ensure safety during testing. The Arduino Uno that is used for low-level control does not have a digital-to-analogue converter, but provides a PWM signal at 7.8 kHz which is then applied to an analogue voltage low-pass filter. To pass this signal to the motor controllers, a Burr-Brown ISO124 isolation amplifier was used along with two MAX680 chips to boost the available 5 V supplies.

IV. Sensor Systems

A. GPS & IMU

The accuracy of standard GPS devices has improved over recent years with the cessation of "Selective Ability", upgrades to the satellite constellation and the introduction of more advanced measurement and augmentation systems [26]. Differential GPS utilizes ground based stations to broadcast GPS error corrections for localized areas and Satellite Based Augmentation Systems such as the US Wide Area Augmentation System perform a similar function using additional geostationary satellites. Real-Time Kinematic systems utilize a local ground station and determine the relative position between the moving object and the ground station via measurement of the GPS carrier phase—such systems are able to achieve centimeter level accuracy [27].

A variety of commercial products are available which provide built-in sensor fusion as well as functionality including D-GPS correction and RTK measurements. The cost is proportional to the accuracy and ranges from around \$1,000 for sub-meter accuracy (e.g. D-GPS), to around \$5,000 for decimeter accuracy (dual channel commercial SBAS) to tens of thousands for advanced survey-grade RTK systems

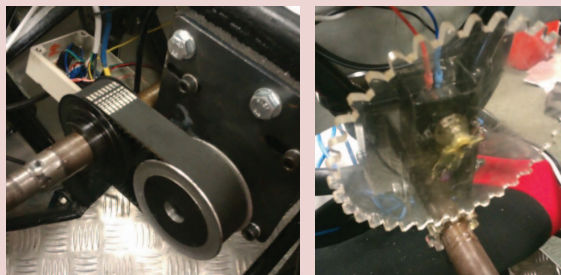


FIG 3 DC motor with drivetrain (left), steering sensor (right).

[27]. Products such as the Applanix POS LV which include these high-level features have commonly been used in autonomous vehicles [24], however, with no public SBAS available in Australia and systems such as D-GPS and RTK still extremely expensive, a standard GPS device was selected for this project. It was found that the jitter of the GPS device used (QStarz BT-Q818X) resulted in a mean deviation from the average position of 1.5 m over a 40 minute test. As a result of this, and the sensor's limited resolution and update rate, it was found to be necessary to improve the performance via sensor fusion.

An Xsens MTi IMU (Inertial Measurement Unit) is used in this project in order to improve the accuracy and frequency of the data provided by the GPS. The unit combines a magnetometer, MEMS 3D accelerometer and MEMS gyroscopes and is designed to output Kalman filtered orientation data for use in the stabilization and control of robots and vehicles as well as calibrated inertial data consisting of acceleration, angular velocity and magnetic field readings. Investigations carried out by manipulating the IMU showed that the accelerometer data is too noisy to integrate directly, with the calculated position diverging in several seconds. The filtered heading data is reasonably useful but tests have shown that it is prone to errors introduced by magnetic field disturbances, particularly when used on the SAE car.

B. Fusion of IMU and GPS Data

Fusion of the IMU heading and GPS track angle (heading) measurements is required due to the speed dependent accuracy of the GPS track angle and the limited absolute and dynamic accuracy of the IMU (see Figure 4). If the car is not moving at an appreciable rate the reported GPS track angle will initially fail to change and jitters substantially at low speeds making it impossible to tell which way the car is facing when starting the autonomous drive. In order to determine the extent of this issue, data was collected by driving in a straight line (directly south) and varying the car's speed and starting/stopping positions on multiple occasions. The resultant deviations from the average angle over the drive were then binned according to speed and standard deviation plotted.

It was observed as expected that the GPS track angle measurement is more reliable at greater speeds. The IMU mean deviation over the same trip was found to be 2.6° however, a considerable risk exists in relying on IMU data as magnetic disturbances can cause result in a loss of absolute accuracy and a "wandering" heading. Whilst the aluminum bracket appears to have minimized the chance of this occurring on the SAE car, the phenomenon was observed several times during testing. A speed-

The filtered heading data is reasonably useful but tests have shown that it is prone to errors introduced by magnetic field disturbances, particularly when used on the SAE car.

dependent weighted average with characteristics based on these observations was therefore implemented in order to ensure that the most accurate heading is always available. The weightings were calculated by applying the method developed by Elmenreich [28], which seeks to minimize the variance of a linear combination (Z) of Gaussian random variables (X_G and X_I) which form the fused estimate:

$$Z = w_G X_G + w_I X_I \quad (1)$$

The weighting factors of the GPS and IMU (w_G and w_I) as well as the combined variance (σ_Z^2) were thus calculated as:

$$w_G = \frac{1}{1 + \frac{\sigma_G^2}{\sigma_I^2}}, w_I = \frac{1}{1 + \frac{\sigma_I^2}{\sigma_G^2}} \Rightarrow \sigma_Z^2 = \frac{1}{\frac{1}{\sigma_G^2} + \frac{1}{\sigma_I^2}} \quad (2)$$

These weightings were calculated for each speed bin and a piece-wise linear relationship between speed and the required weighting ratio was developed and implemented in the control software (see Figure 5). In order to ensure a reliable heading estimate at stationary, the linearized implementation applies zero weighting to the GPS from zero to 0.25 m/s. The most significant result of this approach is the ability to obtain a stable heading throughout the process of acceleration from stationary whilst maintaining absolute accuracy at speed. The effect of reducing the variance of the

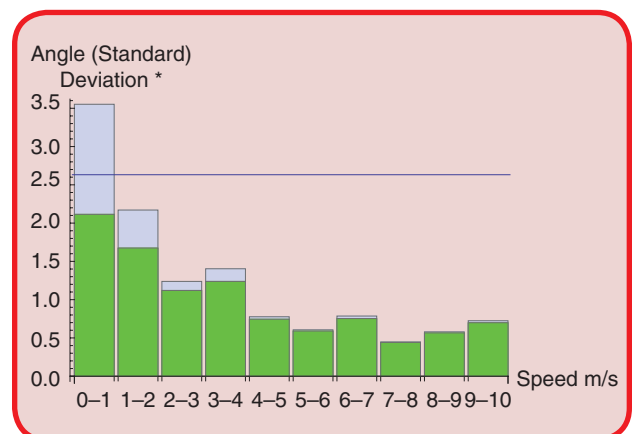


FIG 4 Unmodified GPS (blue) and fused (green) angle deviation vs. speed over a trip with IMU deviation (line).

This system can be considered to be a variation upon a loosely coupled GPS-INS fusion with the acceleration data used directly.

IMU heading at the last GPS fix and an intermediate time to the heading calculated by the weighting formula at the last GPS fix. In this way any bias type error associated with the IMU heading does not affect the interpolated points. At present this method is used to increase the heading update rate to 10 Hz, though higher rates are possible if desired.

heading data through this technique can be noted in the green overlay in Figure 4.

Note that the GPS reports the track angle whilst the IMU outputs the direction the car is current facing. In order to determine the cars trajectory the track angle is required and a correction is made to the IMU heading value based on the angle at which the front wheels are being steered.

There is also a desire for faster heading updates and so the high-rate IMU data is used to interpolate the calculated heading by adding the difference between the steering corrected

A simple filtering algorithm has been implemented in order to smooth and improve the accuracy of the car's positioning data. The concept of applying such a filter to a Global Positioning System/Inertial Navigation System combination is based upon the observation that GPS error, though bounded, is large and may have reliability issues whilst INSs possess a small error over small time-scales and are not subject to the same error sources as GPS [29]. This system can be considered to be a variation upon a loosely coupled GPS-INS fusion with the acceleration data used directly [30] and was chosen over more complex sensor fusion algorithms such as [31] due to the low computational loading, simplicity of implementation and applicability to the low cost hardware employed in this project.

Firstly the acceleration data is transformed into a North-East-Down coordinate system using the orientation data (read as a cosine matrix from the IMU) and then combined in a Kalman Filter with the GPS (Doppler) velocity information. The estimated velocity from this filter is then combined in another Kalman Filter with the position data, which is then used to compute the cars trajectory. Both filters currently use a linear free body model for the system. Initial tests have shown that this method is reasonably effective given its simplicity however future work on this project is expected to focus on implementing an improved model for the vehicle's dynamics and comparing the use more advanced GPS-INS fusion techniques.

C. Laser Scanner

The LIDAR system used in this project is an IBEO Lux automotive LIDAR. The IBEO sensor features four layer

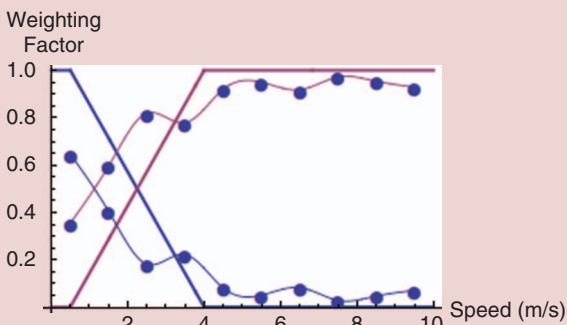


FIG 5 Calculated and linearized weighting factors.

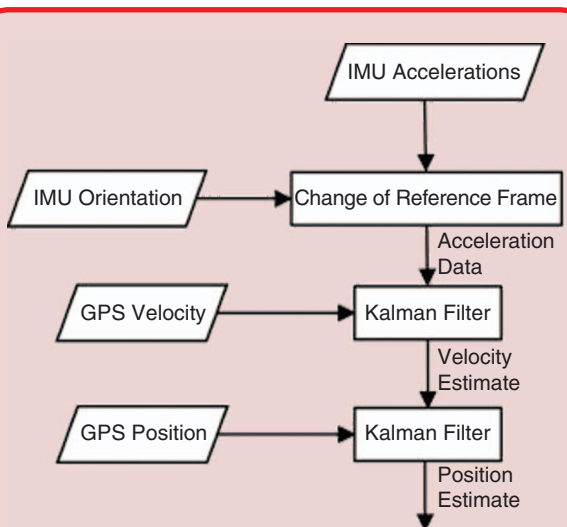


FIG 6 Position fusion algorithm.

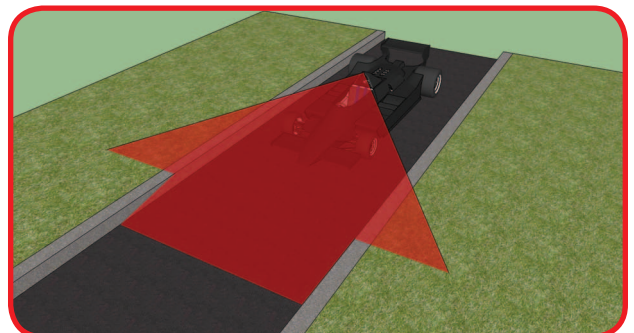


FIG 7 LIDAR edge detection concept.

measurements and has internal data processing functionality including object detection and classification. The data is delivered using TCP/IP over an Ethernet connection and includes scan data in polar coordinates and object data in x - y coordinates referenced to the sensor.

LIDAR scan data is used to detect the road edges by determining the horizontal extent of the point, which make up the roads surface. The sensor arrangement is shown in Figure 7 and is based on the principle that road points on a bitumen surface tend to be arranged collinearly with a small deviation whereas points belonging to uneven surfaces (such as grass and curbs) tend to be scattered and their arrangement depends on the contour of the surface away from the road edge. The IBEO sensor also provides object data which is projected onto the cars running map data as “fence posts” in order to mark obstructions in the path. The graph below shows the range of obstructions (red circles) detected while parked in the laboratory overlaid on the raw scan data.

LIDAR scan data is used to detect the road edges by determining the horizontal extent of the point which makes up the roads surface.

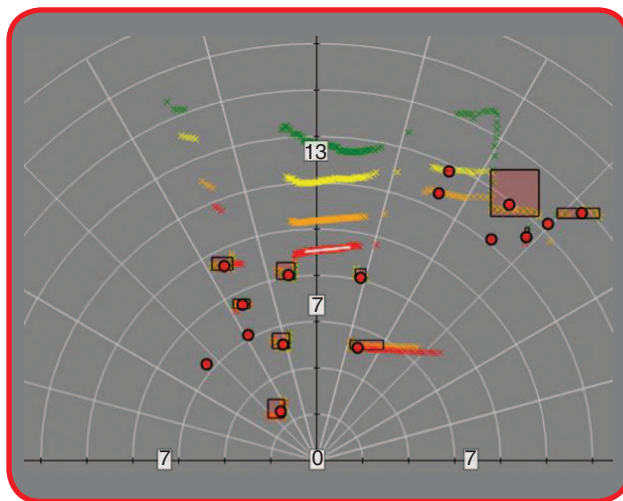


FIG 8 Identification of objects in the car's path.

V. Navigation System

The navigation control system is centered around a small PC running Linux located on the car. An Ethernet switch located in the car provides connectivity for the IBEO sensor and Wi-Fi link to the base station with all other sensors and outputs communicating using serial over USB interfaces. The base station hardware IO software is used to generate a safety “heartbeat” signal which is able to be interrupted with a large physical stop button and allows configuration of control system parameters. The autonomous driving features such as map collection and selection as well as information outputs are provided in a web interface, which communicates asynchronously with the main control system utilizing AJAX techniques which provide a means for communication between JavaScript running in the web-browser and server-side scripts which translate between the web server and main control program.

A. Trajectory Calculation

Maps are captured either by computer input or by actually driving the course. They are stored in text files and contain latitude and longitude coordinates for the maps “datum” followed by waypoints and “fence posts” in Cartesian coordinates

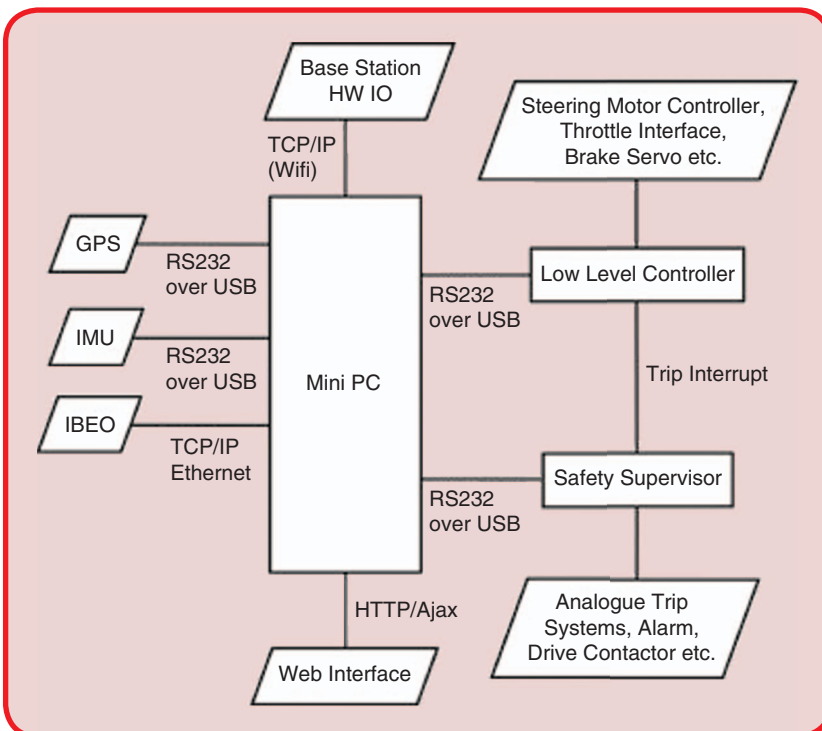


FIG 9 System design overview.



FIG 10 Waypoint data recorded automatically.

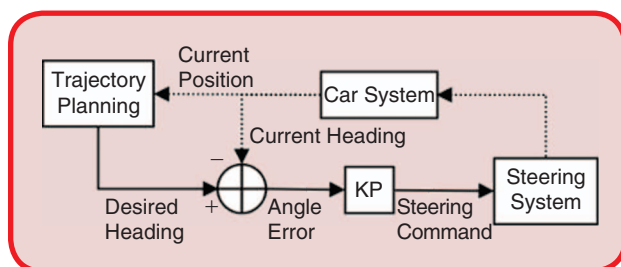


FIG 11 Steering controller overview data recorded automatically.

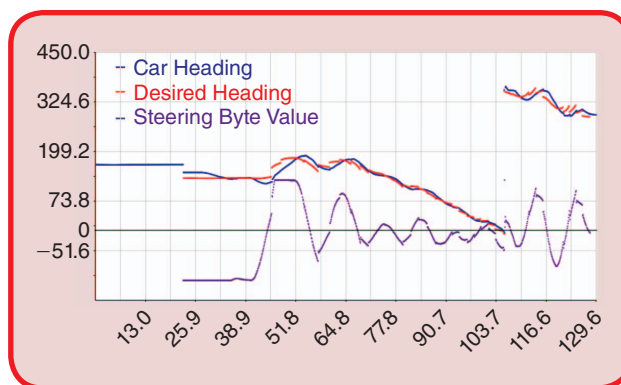


FIG 12 Performance of heading controller during driving.

with distances in meters referenced to the datum. During recording and following the map is loaded into RAM and can be edited dynamically within the control program. Fence posts represent obstructions or boundaries which the car is not allowed to travel near to, and will initiate an emergency stop should the car come within a fixed distance of them. The desired speed is set based on the current turn radius given slower speeds when cornering.

The car's trajectory is calculated each time an updated position measurement is received and a cubic spline is calculated from the current position through the next several waypoints. This allows for generation of a realistic, smooth, trajectory and solves stability issues that would occur should the car not arrive at each waypoint facing a direction reasonable for continuing the required path. The third order polynomial splines used are defined piece-wise, parametrically, between the cars current position, the first waypoint and each successive waypoint.

The splines are required to be fitted against time as in the case of a general map the y coordinate is not a function of x and because the motion of the vehicle is required to be smooth in time as well as in space. In order to achieve this, a "pseudo-time" scale is developed based upon the assumption of constant velocity between the waypoints. The required bearing is then calculated and used as the set point for a PID loop. This PID loop compares measurements of the car's current heading (from the fusion algorithm) to the heading required by the trajectory and manipulates the steering angle in order to bring the cars heading in line. By tuning the PID controller parameters factors such as the angular velocity of turning and response to changing trajectories are easily adjusted. Figure 12 shows the operation of the controller during a test drive—the blue car heading and red desired heading overlay well as required, however discontinuities are present each time the car reaches a waypoint.

The control program is able to either operate the brake or throttle at any one time. The value of the brake travel and throttle value required are computed by two separate PID controllers. This enables tuning of each action independently and is required given the nature of operation of the two functions. Braking occurs in response to a high-speed condition and the brake is operated only until the desired and actual speeds match. Manipulation of the throttle is incremental, since the PID controller is required to converge on a non-zero throttle value, thus the PID output is summed with the current throttle set position to give the new set position at each step.

B. Safety Systems

Safety is a major concern for drive-by-wire and autonomous systems, so additional, independent safety systems

have been introduced in order to increase driver and bystander safety. In addition to the previously mentioned low-level safety systems, which will automatically disable autonomous control whenever steering or brake are operated manually, we have also implemented a high-level safety system, centered around a heartbeat system which ensures that the vehicle can be remotely stopped at all times. As our vehicle has an electric drivetrain, the outputs of the safety systems are simple electrical interlocks which manipulate the drive-by-wire system or shutdown power as required.

Additional functionality includes:

As our vehicle has an electric drivetrain, the outputs of the safety systems are simple electrical interlocks which manipulate the drive-by-wire system or shutdown power as required.

- Safe “arming” sequence ensuring that the car will not accelerate immediately after drive-mode is engaged.
- Notification of control program of any trip conditions detected by the car’s analogue safety interlocks.
- Disconnection of power to the drive system in a trip condition.
- Acoustic notification of trip conditions.
- Activation of emergency braking via a hardwired interrupt to the low level controller.

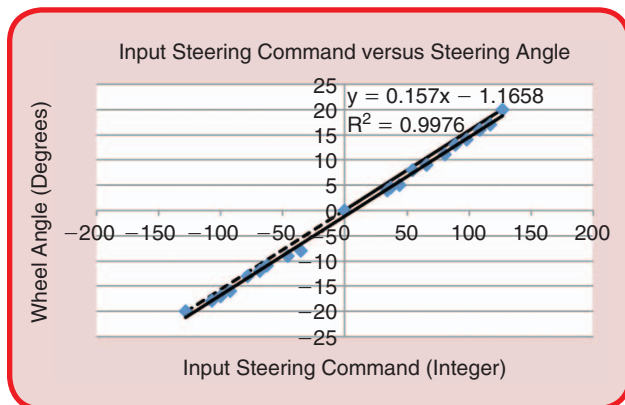


FIG 13 Measuring the linearity of the autonomous steering system and vehicle heading. The dashed line represents the ideal, linear relationship.

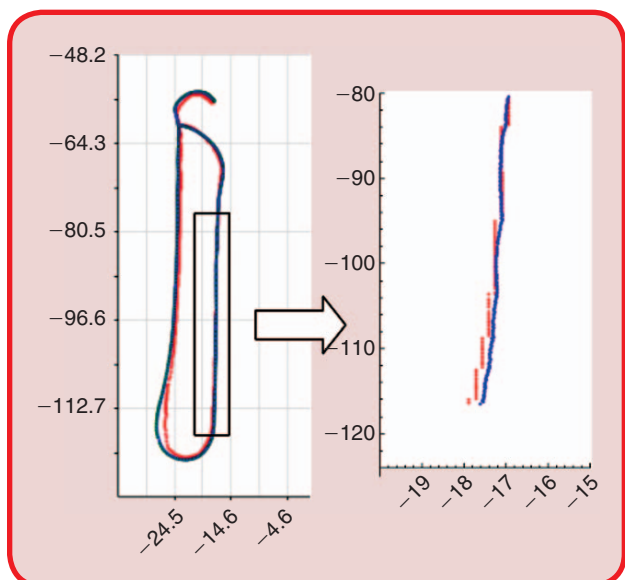


FIG 14 Manually driven path. GPS position (red) and, fused position (blue) in metres with magnification.

VI. Experiments and Results

For the steering system we tested the required speed of the DC motor to turn the steering column, the ability of the system to return to the same position, and the linearity of the low-level controller input to vehicle steering angle.

With no driver in the seat, the speed corresponds to an average of 190°/second. The relationship between input steering commands and the angle that the wheels turn was shown to be extremely linear via data in Figure 13 and allows easy modeling of the drive-by-wire dynamics for control purposes.

In order to verify the operation of the sensor fusion algorithms, a series of tests were conducted where a loop consisting of two straight sections aligned North-South (marked on the ground) was driven. This experiment reveals the effectiveness of the position-filtering algorithm, particularly when expanded in scale as shown to the right in Figure 14. The GPS position resolution and update rate are somewhat limited, whilst the fused position is substantially smoother and masks the GPS module’s shortcomings. The GPS data

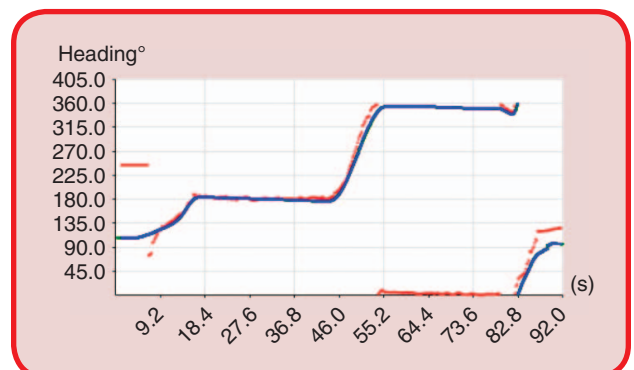


FIG 15 Heading over time for the path shown in the previous figure.

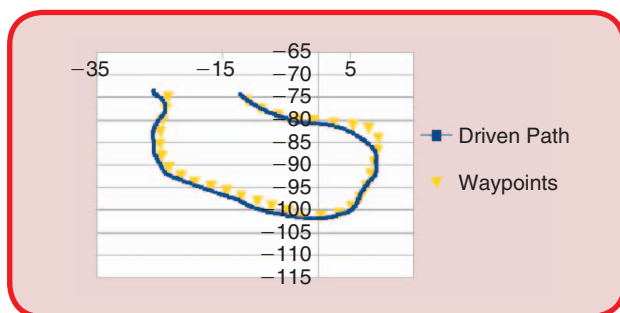


FIG 16 Autonomous driving trajectory in metres.

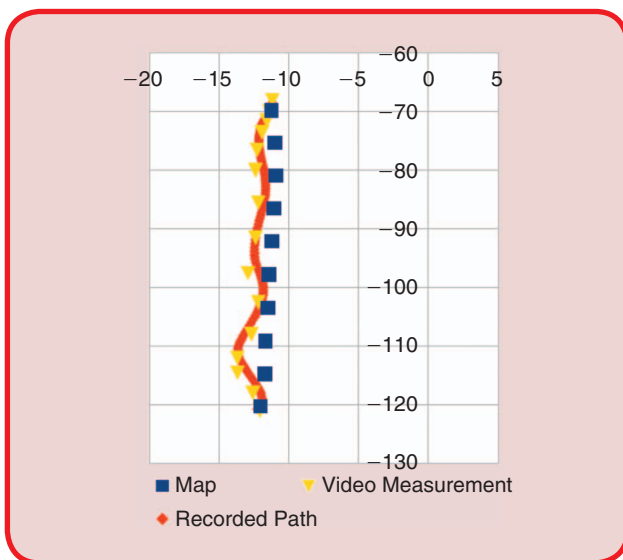


FIG 17 Straight line test path in metres.

around the South-North turn in this test drive shows a deviation of approximately two meters as well as increased noise, however, the filtered position in this region is smooth and more importantly physically consistent with the pattern being driven in the turn. It is therefore evident from this data that the filtering algorithm is of benefit.

The accuracy of the heading fusion algorithm was measured in the same experiment and verified against a magnetic compass. The heading measurements shown in Figure 15 show that the bearing reported by the car's sensors was accurate, with 180° and $0/360^\circ$ measured for the two sections. Evidence of the GPS module's noise, poor resolution and limited accuracy at the start/end of the driving section were also observed in this experiment.

Experiments have shown excellent performance of the overall navigation system via accuracy measurements conducted on a field. A test was conducted by first automatically recording a path when manually driving, then autonomously driving the same path, using a waypoint radius of 2.5 m. The results are shown in Figure 16 with a mean accuracy (distance from path to waypoint) of 80 cm and a closest approach of 10 cm. Note that the "cutting" of the corner at the top right

of the path is due to the combination of the radii of the waypoints in that section.

Further analysis was performed by autonomously driving an automatically recorded map which followed a line marked on the ground. Video analysis, with a camera positioned along the line, was used to measure the deviation of the car from the line and compared to the deviation measured by the vehicle's positioning system (see Figure 17). It was found that a mean lateral error of 70 cm was experienced and that the position reported by the sensor fusion algorithm agreed well with that measured independently, but that the steering response could be improved to address the cyclic deviation observed.

VII. Conclusion

This project shows significant promise as an accessible platform for the development of autonomous driving technology and driver assistance functions. The system has been tested under autonomous control and it was found that the steering system is accurate enough to steer the Formula SAE vehicle in a straight line and to turn with a rotational speed comparable to a human driver. The electronically controlled brake and acceleration functions have shown to operate as expected and with a performance that exceeds the requirements of this project.

Performance of the navigation control system has been validated successfully. The modular design of the software provides the ability to easily modify or implement new functionality and the sensor array has proven to be useful in driving in a variety of scenarios. The safety systems and user interface implemented have provided safe operation whilst actually increasing flexibility during testing. In addition to possible improvements to the filtering techniques used in the sensor fusion, future extensions will involve more accurate positioning sensors in combination with advanced sensor fusion and path planning algorithms. The goal of this work is to optimize driving behaviors and reliability so that autonomous operation at race speed over long distances can be achieved.

Acknowledgments

The authors would like to thank Galaxy Resources and Swan Energy for their generous donations for the REV Project. Thomas Drage acknowledges the assistance of Western Power through their undergraduate scholarship programme.

About the Authors



Thomas Drage has received a B.Eng. degree with honours in Electrical & Electronic Engineering and B.Sc. in Physics from The University of Western Australia in 2014. He has held several internships in the field of Instrumentation and Control and is currently working as a

contractor in the Oil & Gas industry. His research interests include autonomous vehicles, in particular functional safety and advanced sensor systems.



Jordan Kalinowski has received a B.Eng. degree with honours in Electrical & Electronic Engineering from The University of Western Australia in 2013. He currently holds a position with an international engineering services firm and his research interests are in embedded systems and motor control algorithms.



Thomas Bräunl (M'97-SM'08) has received a Diploma degree in Informatics from Universität Kaiserslautern, Germany, the M.S. degree in Computer Science from University of Southern California, Los Angeles, CA, USA, and the Ph.D. and Habilitation degrees in parallel processing from Universität Stuttgart, Germany. He is currently Professor in Electrical and Computer Engineering at The University of Western Australia, Perth, Australia. He has worked in Germany for Mercedes-Benz, BMW, and BASF and has held guest professor appointments at TU München, Germany, and Santa Clara University, CA, USA. He is creator of the EyeBot mobile robot and embedded controller family, and directs the Renewable Energy Vehicle Project (REV) at UWA, where so far four electric vehicles and two autonomous vehicles have been constructed, and a city-wide charging network has been established. *Embedded Robotics* (Springer 2008) and *Parallel Image Processing* (Springer 2001) are two of his major publications in book form.

References

- [1] Society of Automotive Engineers. Formula SAE—FSAE.com. [Online]. Available: <http://www.fsae.com>
- [2] T. Black, "A driver assistance system for a BMW X5," M.S. thesis, Univ. Western Australia, Perth, Australia, 2012.
- [3] M. Claudia, S. Felicia, B. Karoly, and T. Raluca, "PMSM design for brake-by-wire technology in automobiles," *J. Comput. Sci. Control Syst.*, vol. 5, no. 1, pp. 237–240, May 2010.
- [4] N. A. M. Azubir, M. K. Hassan, H. M. I. Nizam, B. S. K. K. Ibrahim, and S. F. Toha, "Optimal design of electric power assisted steering system using GA-PID method," in *Proc. Int. Symp. Robots Intelligent Sensors*, 2012, pp. 614–621.
- [5] R. Marino, M. Netto, and S. Scalzi, "Integrated driver and active steering control for vision-based lane keeping," *Eur. J. Control*, vol. 18, no. 5, pp. 475–484, 2012.
- [6] M. B. Baharom, A. J. Day, and K. Hussain, "Design of full electric power steering with enhanced performance over that of hydraulic power-assisted steering," *Proc. Inst. Mech. Eng., Part D: J. Automobile Eng.*, vol. 227, no. 3, pp. 590–599, Mar. 2013.
- [7] T. A. Coombs, N. Lii, and S. Sturm, "Driver-input sensor selection and topologies for fault-tolerant drive-by-wire applications," in *Proc. IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics*, Monterey, CA, July 2005.
- [8] M. Gunzert and A. Nägele, "Component-based development and verification of safety critical software for a brake-by-wire system with synchronous software components," in *Proc. Int. Symp. Software Engineering Parallel Distributed Systems*, May 1999.
- [9] R. Sartori, S. M. Savaresi, and M. Tanelli, "Combining slip and deceleration control for brake-by-wire control systems: A sliding-mode approach," *Eur. J. Control*, vol. 13, no. 6, pp. 593–611, 2007.
- [10] C. Rossi, A. Tili, and A. Tonielli, "Robust control of a throttle body for drive by wire operation of automotive engines," *IEEE Trans. Control Syst. Technol.*, vol. 8, no. 6, pp. 995–1002, Nov. 2000.
- [11] C.-H. Chen, H.-L. Tsai, and Y.-S. Lin, "Servo control design for electronic throttle valve with nonlinear spring effect," in *Proc. 11th IEEE Int. Workshop Advanced Motion Control*, Mar. 2010, pp. 88–93.
- [12] A. Shaout, D. Colella, and S. Awad, "Advanced driver assistance systems past, present and future," in *Proc. 7th Int. Computer Engineering Conf.*, 2011, pp. 72–82.
- [13] M. Muffert, D. Pfeiffer, and U. Franke, "A stereo-vision based object tracking approach at roundabouts," *IEEE Intell. Transp. Syst. Mag.*, vol. 5, no. 2, pp. 22–32, Apr. 2013.
- [14] E. D. Dickmanns, "The development of machine vision for road vehicles in the last decade," in *Proc. Intelligent Vehicle Symp.*, 2002, pp. 268–281.
- [15] I. Miller, S. Lupashin, N. Zych, P. Moran, B. Schimpf, A. Nathan, and E. Garcia, "Cornell university's 2005 DARPA grand challenge entry," *J. Field Robot.*, vol. 25, no. 8, pp. 625–652, Aug. 2006.
- [16] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrosseck, C. Koelen, C. Markey, C. Rummel, J. van Niekirk, E. Jensen, P. Alesandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney, "Stanley: The robot that won the DARPA grand challenge," in *The 2005 DARPA Grand Challenge*, M. Buelher, K. Iagnemma, and S. Singh, Eds., Berlin Heidelberg, Germany: Springer-Verlag, 2007, pp. 1–43.
- [17] A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, C. Reinholdt, D. Hong, A. Wicks, T. Alberi, D. Anderson, S. Cacciola, P. Currier, A. Dalton, J. Farmer, J. Hurdus, S. Kimmel, P. King, A. Taylor, D. van Covern, and M. Webster, "Odin: Team VictorTango's entry in the DARPA Urban Challenge," *J. Field Robot.*, vol. 25, no. 8, pp. 467–492, Aug. 2008.
- [18] M. Montemerlo, J. Becker, H. Dahikamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhne, D. Johnson, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, and M. Pflueger, "Junior: The Stanford entry in the urban challenge," *J. Field Robot.*, vol. 25, no. 9, pp. 569–597, 2008.
- [19] J. Bohren, T. Foote, J. Keller, A. Kushleyev, D. Lee, A. Stewart, P. Ver-naza, J. Derenick, J. Spletzer, and B. Satterfield, "Little Ben: The Ben Franklin Racing Team's entry in the 2007 DARPA Urban Challenge," *J. Field Robot.*, vol. 25, no. 9, pp. 598–614, Sept. 2008.
- [20] A. Broggi, P. Medici, P. Zani, A. Coati, and M. Panciroli, "Autonomous vehicles control in the vislab intercontinental autonomous challenge," *Annu. Rev. Control*, vol. 36, no. 1, pp. 161–171, Apr. 2012.
- [21] J. Markoff. (2013, Aug. 17). Google cars drive themselves, in traffic. *New York Times*. [Online]. Available: <http://www.nytimes.com/2010/10/10/science/10google.html>
- [22] Electronic News Publishing, "Komatsu and Rio Tinto enter into agreement for 150 autonomous truck deployment into pilbara iron ore operations by 2015," *ENP Newswire*, 4 Nov. 2011.
- [23] B. Dodson. (2012, Nov. 11). Autonomous Audi almost matches veteran race car drivers' lap times. [Online]. Available: <http://www.gizmag.com/stanford-audi-tts-autonomous-car-thunderhill/24957/>
- [24] T. Brown, "Handling at the limits," *GPS World*, vol. 21, no. 8, pp. 38–41, Aug. 2010.
- [25] R. Granger and R. S. Sharp, "On car steering torques at parking speeds," *Proc. Inst. Mech. Eng. Part D: J. Automobile Eng.*, vol. 217, no. 2, pp. 87–96, Feb. 2003.
- [26] National Coordination Office for Space-Based Positioning. Navigation and Timing. (2013, June 24). GPS accuracy. [Online]. Available: <http://www.gps.gov/systems/gps/performance/accuracy/>
- [27] USU/NASA Geospatial Extension Program. (2010, Sept. 3). High-End DGPS and RTK systems. [Online]. Available: http://extension.usu.edu/nasa/files/uploads/gtk-tuts/rtk_dgps.pdf
- [28] W. Elmenreich, "Fusion of continuous-valued sensor measurements using confidence-weighted averaging," *J. Vib. Control*, vol. 13, nos. 9–10, pp. 1305–1312, Sept. 2007.
- [29] G. Falco, G. A. Einicke, J. T. Malos, and F. DAVIS, "Performance analysis of constrained loosely coupled GPS/INS integration solutions," *Sensors*, vol. 12, no. 11, pp. 15983–16007, Nov. 2012.
- [30] H. Qi and J. B. Moore, "Direct Kalman filtering approach for GPS/INS integration," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 2, pp. 687–695, Apr. 2002.
- [31] Y. Li, J. Wang, C. Rizos, P. Mumford, and W. Ding, "Low-cost tightly coupled GPS/INS integration based on a nonlinear Kalman filtering design," in *Proc. National Technical Meeting Institute Navigation*, Monterey, CA, Jan. 2006, pp. 958–966.

CHAPTER 2

LIDAR Road Edge Detection by Heuristic Evaluation of Many Linear Regressions

LIDAR Road Edge Detection by Heuristic Evaluation of Many Linear Regressions

Thomas Drage¹, Thomas Churack¹ and Thomas Bräunl¹, *Senior Member, IEEE*

Abstract—This article presents a novel algorithm for the reliable detection of road edges via multi-layer LIDAR sensing on a moving vehicle for improved safety. Results are presented showing the operation and performance of the system in several outdoor scenarios. The algorithm is then integrated with orientation and position measurements, giving improved reliability and extension to mapping.

Keywords—LIDAR, mapping, computer vision, autonomous vehicles.

I. INTRODUCTION

The Renewable Energy Vehicle (REV) Project at the University of Western Australia conducts research into electric vehicles, vehicle automation and autonomous driving systems. Recent projects include the development of an Autonomous Formula-SAE Electric car [1]. This vehicle is an open-wheeled, electric drive race car, with electronic drive-by-wire and electromechanical brake/steering actuation. The vehicle serves as a compact, flexible test-bed for sensor testing and the development of autonomous driving algorithms. In this project, the motivation for investigating LIDAR road-edge detection is twofold; for safety (an independent check of position on the road) and for mapping.

Prior research has been conducted on road and road edge detection through optical systems [2, 3, 4], radar [5, 6, 7] as well as through the use of Light Density and Ranging (LIDAR) sensors such as in the winning entry in the 2007 DARPA Urban Challenge [8] and in research at German [9] and Singaporean [10] universities. The methodology described in [7] utilises a feature-extraction algorithm while other algorithms such as [10, 11, 12] rely on the presence of curbs and seek to identify and track curbs as features in the LIDAR data. The approach described in [9] is similar to the initial algorithm employed here and attempts to seek appropriate linear fits to the road surface, however their estimates were then combined with filtered reflectivity data in order to determine the road surface. In both of the latter two cases a Kalman filter is used to track the position of the road edges over time.

It has been shown that the accuracy of LIDAR based mapping can be improved by adjusting incoming LIDAR data for the sensor's pitch and roll, as measured by an onboard Inertial Measurement Unit (IMU) [13, 14, 15]. This strategy has been proven to be viable even for unmanned aerial vehicles [15], which require compensation for a much greater degree of pitch, roll and yaw than an autonomous land vehicle is likely to. By analysing LIDAR data consisting of multiple scan layers intersecting the roadway, an estimation of road shape and road curvature can be calculated [15, 16].

The problem of path-finding can be described as: “Given a start state, a goal state, a representation of the robot and a representation of the world, find a collision free path that connects the start with the goal satisfying the system

constraints” [17]. In mobile robotics a proven method to obtain the requisite “representation of the world” is via the use of LIDAR data to generate a virtual map in real-time both as the sole sensor [18] and in conjunction with data from additional sensors [19, 20]. Similar LIDAR based map building approaches have been shown to be suitable for outdoor terrain [21] and have been scaled up to larger autonomous vehicles with great success [22, 23]. These generated maps vary from simple two-dimensional maps suitable for basic path planning consisting of traversable regions, obstacles and unexplored regions [24, 25] to more complex three-dimensional maps from which sophisticated cost maps are generated [23, 26].

II. BACKGROUND

The LIDAR system used in this project consists of an IBEO Lux automotive LIDAR. This sensor utilises reflected infra-red light in order to measure distance (via time-of-flight) and can build a 3D point cloud by scanning horizontally in four vertical layers. The IBEO sensor has sophisticated internal data processing functionality including object detection and classification. Data is delivered using TCP/IP over an Ethernet connection and includes scan data in polar coordinates and object data in x-y coordinates referenced to the sensor.

TABLE I. LIDAR CHARACTERISTICS

Specification	Value
Technology	Time of flight (output of distance and echo pulse width)
Range	200m
Field of View (Horiz / Vert)	85° / 3.2°
Layers	4
Echo Detection	3 measurements per pulse
Update Rate	Up to 50Hz
Accuracy	10cm

In this project the sensor was mounted on a specially constructed bracket above the car's roll cage. Vertical angle adjustment is provided so that the sensor can be positioned and locked in the optimal orientation; slightly below the horizontal when used to determine the position of the road. The mounting is secured to the chassis in three points in order to provide a stiff mounting and minimise vibration while the sensor itself is mounted on a piece of waterproof fibreboard and attaches to the frame via saddle clamps attached to the top cross-bar. Locking segments located at the bottom edges prevent the sensor angle from changing due to vibrations whilst driving.

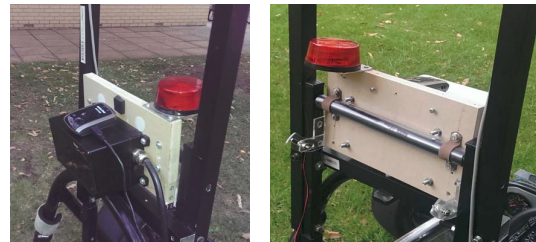


Figure 1: LIDAR mounting

¹ The authors are with The REV Project at The University of Western Australia, <http://therevproject.com>, thomas.drage@research.uwa.edu.au

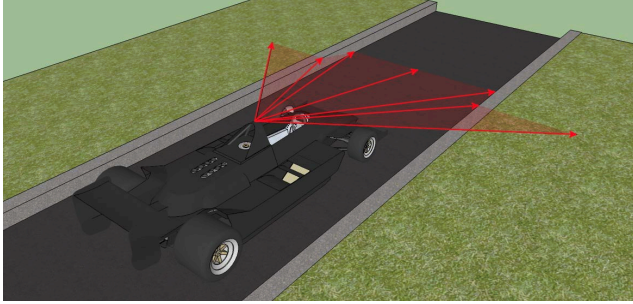


Figure 2: LIDAR edge detection geometry.

In this project the IBEO sensor's "scan data" sets consisting of polar angle/radial distance pairs were analysed in order to find the horizontal extent of the road surface. By mounting the sensor at height (i.e. above the roll cage) the sensor plane intersects the road in such a way that radial distance variations are recorded for both variations in the height of the road/curb as well as in the distance in front of the car (see Figure 2). It was found during experimentation with the IBEO sensor that road points on a bitumen surface tend to be arranged co-linearly with a small deviation, whereas points belonging to uneven surfaces (such as grass and curbs) tend to be scattered, with their arrangement depending on the contour of the surface away from the road edge.

In the case of the Autonomous SAE car, detection which relies upon the presence of curbs or marked lines is not feasible as race tracks, unlike public roads, may not have such features. In addition, the algorithm must be able to dynamically detect road edges whilst the car is moving (and turning) and should have an accuracy better than 0.5m at each side to allow a safe margin between the vehicle and the road edge. As physical road edges are often poorly defined, it is not desirable for the vehicle to be allowed closer than this distance. The algorithm

developed here therefore focuses on identifying the presence of a road section rather than the presence of a particular edge characteristic and will thus work both on curbed and non-curbed roads.

III. HYPOTHESIS FOR LINEAR REGRESSION

The following hypothesis was developed for the detection of road edges:

Roads should be:

- Mostly, but not always close to the center of the scan
- Smooth (e.g. points co-linear)
- In the same plane as the car (e.g. small horizontal gradient)

Edges may be:

- Scattered (non-linear)
- Sloped
- Raised/lowered

An additional property, considered later, is that the road-edge boundary should be locatable in a predictable fashion over time.

The algorithm implemented in this project operates by first identifying a candidate group of points close to the center of the scan data which meet the slope condition (i.e. the slope of a line through these points is less than a pre-set value). This group is then expanded iteratively, with a least-squares linear regression performed at each step. By minimising the square residuals between the fit line (y) and the data (x_i, y_i), this technique obtains the most appropriate line for the given data set, the success of which can be measured by means of the product-moment correlation coefficient r . In this case, the slope (b) and r^2 values are of relevance and are tabulated.

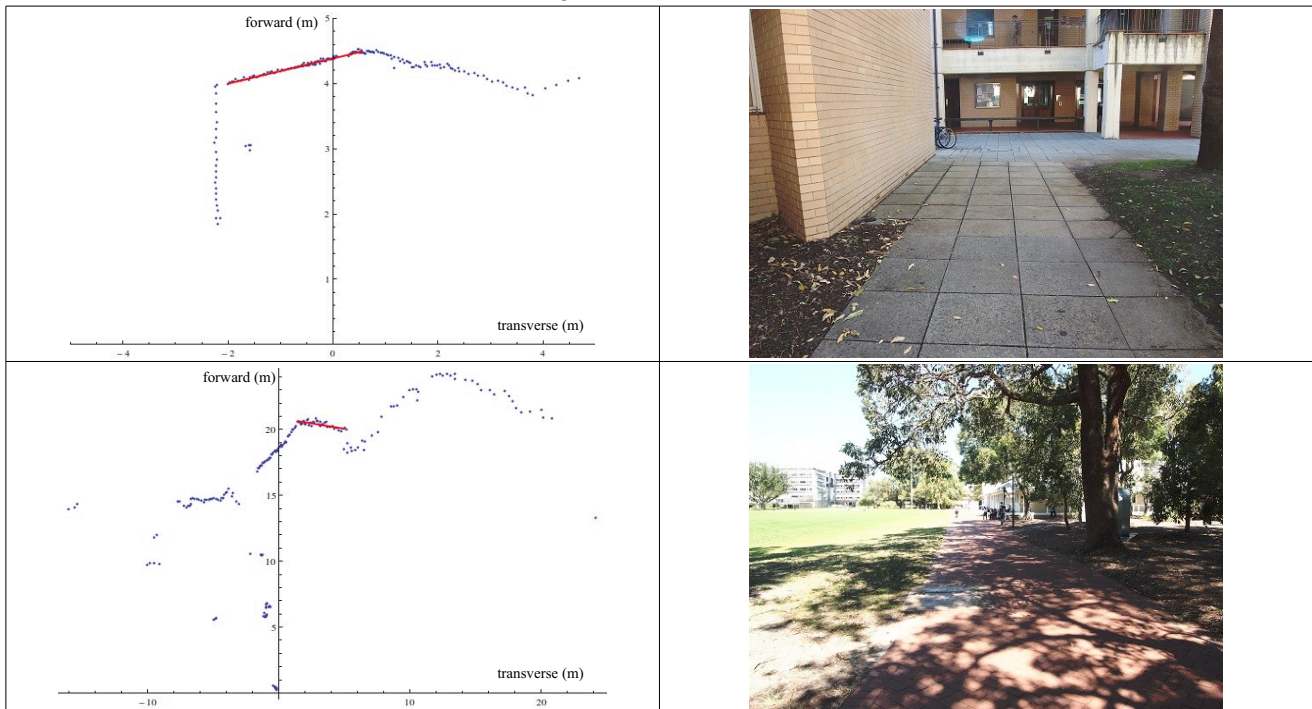


Figure 3: Basic LIDAR edge detection using Linear Regression – distance in metres, looking forward from the vehicle.

Specifically; $y = (\bar{y} - b\bar{x}) + bx$, $r = \frac{s_{xy}}{s_x s_y}$ where

$$b = \frac{s_{xy}}{s_x^2}, \quad s_{xy} = \frac{\sum_{i=1}^n x_i y_i}{n} - \bar{x}\bar{y}, \quad s_x^2 = \frac{\sum_{i=1}^n x_i^2}{n} - \bar{x}^2, \quad s_y^2 = \frac{\sum_{i=1}^n y_i^2}{n} - \bar{y}^2$$

This technique is performed independently for the left and right hand sides of the candidate point group to allow for the fact that roads are often sloped about the center (e.g. to let water run off) which results in improved accuracy. In the initial implementation, the road-edges were then determined to be the outer edges of the candidate groups which maximised the respective correlation coefficients and met the slope condition. In summary, the algorithm follows this sequence:

1. Step out from the center of the dataset ($x_{i=n/2}$) looking for a point cluster $A = \{x_i, x_{i+GROUPSIZE}\}$ which meets the slope condition $b[A] < MAXSLOPE$. Interleave left/right until found.
2. Perform stepping to the left and right (separately) of this cluster, increasing the size (i.e. let $B = \{x_i, x_{i+GROUPSIZE+j}\}$). Fit lines and record the slope and correlation coefficient ($r^2[B]$) at each step.
3. Road edges are at the point which maximises $r^2[B]$ whilst meeting the slope condition.
4. Calculate overall fit line and check that the slope and correlation conditions are met.

Two examples showing captured scan data with the identified road segment lines and a photograph of the respective scenarios are shown below in Figure 3. In the first example the car is positioned with a wall to the left and a grassed area to the right. The sensor has been angled downwards to give a small field of view with the LIDAR scan plane hitting the ground around 4m from the car. The algorithm has correctly identified the road surface based on the linearity alone, highlighting the variety of edge scenarios supported by this algorithm.

The second example is substantially more challenging and shows a brick path with an undulating section of grass to the right and a garden with trees to the left. The sensor has been positioned to measure the ground some 20m out and so the path occupies a very small section of the horizontal span and the uneven bricks result in scatter which further increases difficulty. The road area has again been successfully identified despite the increased difficulty in this scenario.

IV. HEURISTIC ALGORITHM

It was found, however, that this algorithm did not meet the required performance criteria (lateral error $< 0.5m$) in terms of correct identification of the edge and consistency whilst in motion. As a result, a more advanced algorithm was developed, which has the same basis as the one described above but with some heuristic intelligence added to the identification of the correct edge value based upon the correlation coefficient data. Central to this approach was the implementation of a Kalman filter [41] which is used to create a time averaged estimate of the road-edge position which can then be used to assist in location of the current road edge. The second order Kalman filter implemented has the following state transition equation and observation matrix:

$$\begin{bmatrix} x_n \\ v_n \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{n-1} \\ v_{n-1} \end{bmatrix}, \quad H = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

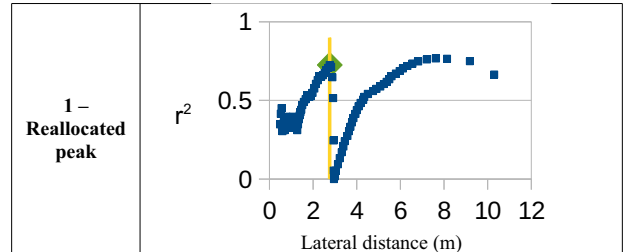
Thus, a position measurement, x , is provided to the filter and the lateral velocity, v , of the road-edge is a quantity derived within the filter. This approach allows for the estimation of the road edge whilst it's horizontal position relative to the car changes, for example when the road or car deviate from a straight line during cornering or "lane changing".

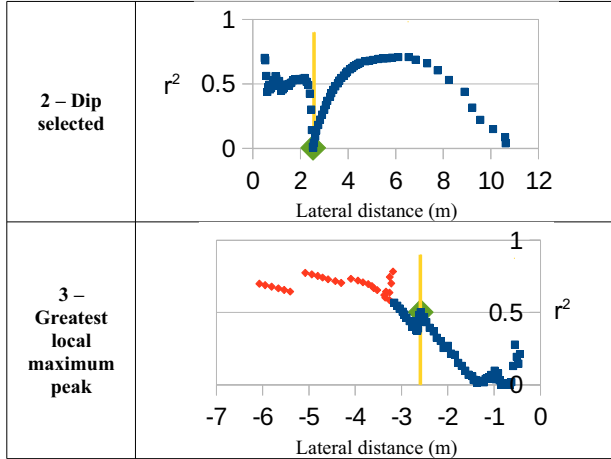
The filtered road-edge position estimate is then used in order to improve the accuracy of the identification of the road-edges from the r^2 - x data set. The local maxima are identified using the criteria: $\max(x) = x_i > x_j, \forall x_j \in [x_{i-k/2}, x_{i+k/2}]$ where k is the number of points that a maximum must exceed and is conveniently set to the size of the candidate groups used initially in the algorithm. This information allows a more appropriate maximum point to be selected if desired and also allows selection of the value which makes the road segment largest should the absolute maximum peak be rather broad. It was also observed that much of the time a significant "dip" in correlation was observed at the point when the group was expanded to begin to include features outside the road-edge, giving a second indicator for the position of the road-edge. Thus, four modes for selection of the road-edge were implemented and a pre-set quantity the allowed deviation introduced:

1. If the value that gives the absolute maximum is different by a quantity exceeding the allowed variation, attempt to find the local maximum closest to the current estimate. If it is not possible to find a clear local maximum, attempt mode 2.
2. If the value that gives the absolute maximum is different by a small amount, but less than the allowed deviation and is to the inside of the absolute minimum, return the value of the absolute minimum correlation.
3. If the greatest local maximum peak meets the minimum correlation requirement, return the point at which it is located.
4. If it was not possible to find a reasonable edge candidate, return failure.

Examples of cases in which these modes were applied are shown below in Table 2. The yellow lines indicate the current estimate used, the green diamonds the identified road-edge point and red points exceed the slope condition.

TABLE II. DETECTION MODES





The most important pre-set parameters for this algorithm are the slope condition value and allowed variation. The slope condition is set based upon the nature of the road surface and severity of roll experienced – a lower value is preferable and values in the range 0.2-0.3 are typical. The allowed variation is generally set at around 1.5m but may be increased if there is significant expected variation in the road edge position. Limits are placed on the distances both transversely and laterally as to how far away edges should be found and are set based on the sensor orientation. Other parameters including the candidate group size and minimum correlation have been set based upon testing and have been found to be non-critical and do not need any adjustment based upon driving conditions etc. With this improved algorithm fine tuning of parameters is generally not required beyond a first approximation.

V. PERFORMANCE TESTING

In addition to the static proof-of-concept testing in Figure 3, trials were run in order to measure the success of the algorithm in dynamic scenarios. In the experiments shown here, the car was driven down a section of road of fixed width, at an approximately constant speed and the road edges detected at approximately 4 Hz (limited by the test configuration). The absolute accuracy of the algorithm can thus be inferred from the measured width of the road, a measurement which can be considered to be the sum of two random variables $Z = X + Y$, denoting the left and right edge positions respectively. The standard results for combination of Gaussian random variables can be used to infer the variance in the determination of each edge position by measurement of Z and assumption that X and Y are independent when the car is well aligned. Thus:

$$\text{If } Z \sim N(\mu_X + \mu_Y, \sigma_X^2 + \sigma_Y^2) \text{ where } X \sim N(\mu_X, \sigma_X^2), Y \sim N(\mu_Y, \sigma_Y^2) \text{ then } \sigma_Z^2 = \sigma_X^2 + \sigma_Y^2.$$

Now, assuming that X and Y are independent and identically distributed and we have:

$$\sigma_X^2 = \sigma_Y^2 = \frac{\sigma_Z^2}{2}.$$

These result is not surprising and hence an estimate of the error in determining an individual edge can be estimated from the road-width measurements. A similar argument allows us to estimate the difference between the mean of the observed and independently measured data, e.g. $\Delta \mu_X = \Delta \mu_Z = \frac{\Delta \mu_Z}{2}$.

1) Scenario 1 – Curb Detection on an Asphalt Road



Figure 4: Road section used in testing.

In this trial, the car was driven in a straight line along a curbed asphalt road as shown in Figure 4. The edges of this road were therefore linear in time up until around 60s at which point the right hand curb curved off to the right. Figure 5 shows the detected road edges in “raw” form generated by the simple peak detection algorithm as well as the “assisted” identification of more appropriate edges with the four mode classification algorithm and finally the Kalman “filtered” road edge position. The algorithms were configured with a maximum allowed slope of 0.4 and allowed variation of 1.5m.

It was found that the average road width was 5.39m with a standard deviation of 0.23m. The road edge was measured manually in four locations and found to have an average width of 5.59m with a standard deviation of 0.01m. The slightly conservative (3.5% low) estimate for the width is most likely due to the presence of leaf matter and the slight taper present at the edge of the asphalt. The estimated standard deviation of an individual edge is 0.16m and thus in this scenario the edge has been identified with an accuracy of approximately 0.10 ± 0.16 m which certainly meets the stated performance requirement of 0.5m accuracy at each side.

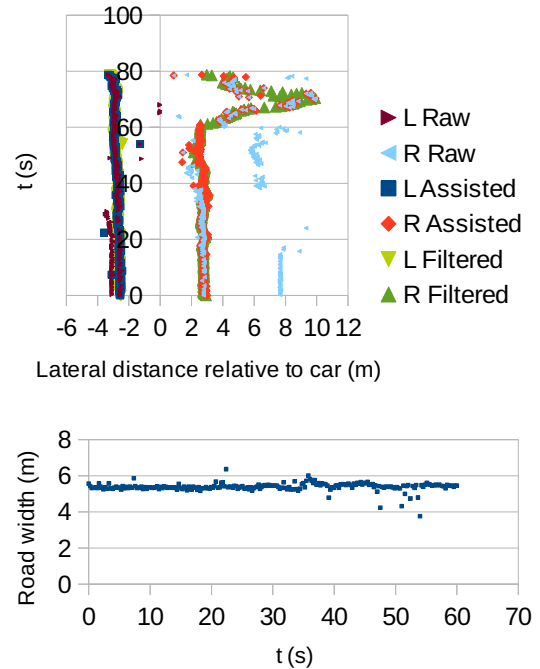


Figure 5: Road edge positions (top) and measured width (bottom) over time for a drive along a curbed asphalt road.

2) Scenario 2 – Grass Edge Detection on a Paved Road

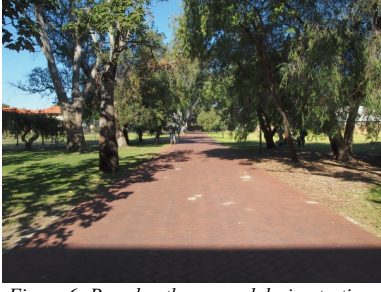


Figure 6: Paved pathway used during testing.

This test is was performed by driving the car down a section of brick paved road with grassed edges as shown in Figure 6. The path was not perfectly straight and due to the lack of clearly defined edge features less accuracy is expected. As a result, the algorithms were configured with a maximum allowed slope of 0.15m and an allowed variation of 1m. Results showing the edge position and measured path width are shown in Figure 7.

It was found that the average road width was 5.16m with a standard deviation of 0.44m. The road edge was measured manually in three locations and was found to consistently measure 5.05m, giving an average error of 2.2%. The estimated standard deviation of an individual edge is 0.31m and thus in this scenario the edge has been identified with an accuracy of approximately 0.06 ± 0.31 m which also meets the stated performance requirement of 0.5m accuracy at each side.

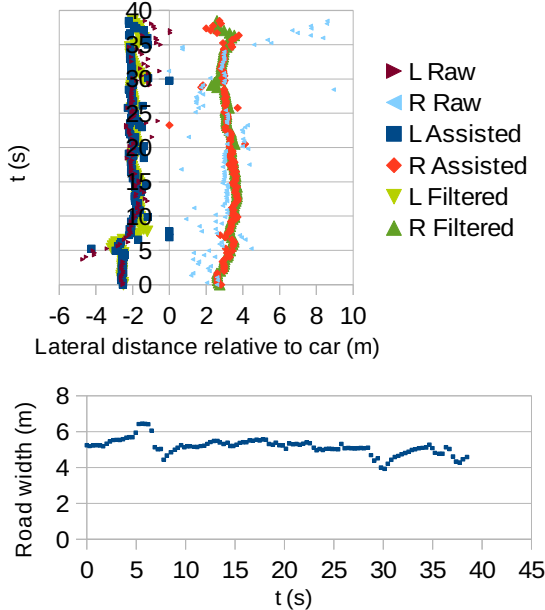


Figure 7: Road edge positions (top) and measured width (bottom) over time for a drive along a paved road with grass edges.

3) Scenario 3 – Curb Detection with Lateral Motion

In this scenario the car was driven along the same road segment as in Scenario 1 above, however the car was in a “wavy” pattern to simulate conditions under which the road

edge position would be constantly changing. For this trial the maximum slope was set at 0.5 and allowed variation at 1.5m.

Due to the constantly changing angle between the car and the road in this test it is not appropriate to draw conclusions from measurement of the road width, however it can be seen qualitatively the algorithm has performed well and is able to track the road edges under dynamic conditions.

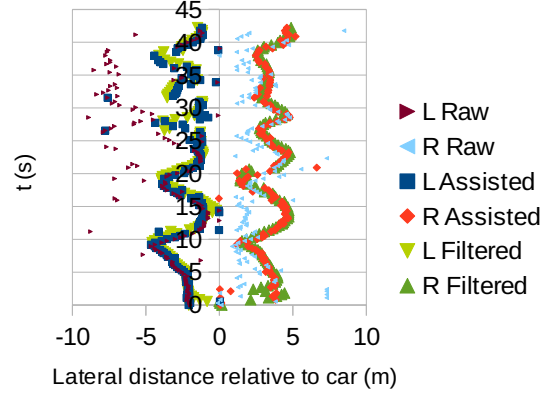


Figure 8: Road edge position on a curbed road with lateral motion.

4) Performance of Edge Finding Modes

It was observed that the simple greatest local maximum peak detection mode was used in the majority of cases in scenarios 1 and 2, with the greater number of favoured dips in scenario 2 most likely due to the different edge characteristic. Pleasingly very few failures were experienced overall, however a large proportion of detected edges in scenario 3 required were identified via the reallocation mode, suggesting that the edges were often not evident without assistance in this scenario. This result supports the need for implementation of temporal filtering in this application as it has evidently improved the edge detection rate very substantially in this project. False positive rates (based on the 0.5m each side performance requirement, i.e. 0.707m width) and known-failure rates are shown below in Fig. 9 also.

Scenario 1	Scenario 2	Scenario 3
False Positive: 3.2% Failures: 0%	False Positive: 12% Failures: 7.7%	False Positive: N/A Failures: 0%

Figure 9: Proportional frequency of edge identification modes and failure rates.

VI. MULTI-LAYER MAPPING WITH STANCE COMPENSATION

The road-finding algorithm can be run over each of the individual scan layers, allowing the vehicle to detect up to as many road segments at varying distances. This allows for deviations in road direction to be detected from further away, while not sacrificing the ability to detect road-edges immediately in front of the vehicle.

As the vehicle drives over a non-flat surface, its pitch and roll angles fluctuate, introducing noise into the Kalman Filter and making it more difficult to track road edges. This error increases with the distance of scan from the vehicle (ie. for higher scan layers). For this reason, correction for the vehicle's pitch and roll are performed prior to the scan information being passed into the road-finding algorithm, minimising what would otherwise be a significant source of noise to the Kalman Filter. The scan data is first adjusted for rotation of the car, converted from a polar coordinate system to a local Cartesian system and then processed by the road finding algorithm. In this case, a six axis IMU attached to the car provides the vehicles pitch and roll relative to the horizontal plane.

The road-finding algorithm returns a pair of edge coordinates in local coordinates. In order to determine the location of road edges and scan points globally, these points must be rotated to account for the vehicle's heading at the time of the scan and the vehicle's current distance from the global origin. The vehicle's heading at any time is calculated from a fusion of IMU and GPS data, generating heading and position data relative to a North-East oriented coordinate set with a nearby predetermined origin. The road edge locations are rotated to account for heading and then projected onto the North-East coordinates relative to the vehicles current position. In this way, a map of the roadway being traversed is generated.

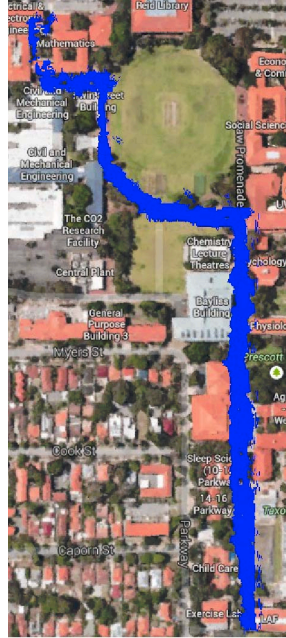


Figure 10: Mapping a paved path.

VII. CONCLUSIONS

We have presented a novel approach to road-edge detection which is applicable both to smooth curbed roads and complex scenarios featuring uneven roads with poorly defined edges. The algorithm presented is applicable both to road-keeping for the safety of autonomous vehicles and for mapping the vehicle's surroundings. We have shown single-dimensional and two dimensional cases as well as the use of multi-layer LIDAR data, IMU information and GPS coordinates to build a two-dimensional map of a road based edges seen along the vehicle's path of travel. Future work will focus on testing in the intended "race track" conditions as well as integrating this algorithm with the vehicles control and path planning functionality.

REFERENCES

- [1] T. Drage, J. Kalinowski and T. Bräuml, "Integration of Drive-by-Wire with Navigation Control for a Driverless Electric Race Car". *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 4, pp. 23-33, Oct. 2014.
- [2] J. Seigemund, U. Franke and W. Forstner, "A temporal filter approach for detection and reconstruction of curbs and road surfaces based on Conditional Random Fields". *2011 IEEE Intelligent Vehicles Symposium*, pp 637-642, Jun. 2011.
- [3] C. Oh, J. Son and K. Sohn, "Illumination robust road detection using geometric information", *2012 15th International IEEE Conference on Intelligent Transportation Systems*, 2012.
- [4] P. Shinzato, D. Gomes and D. Wolf, "Road estimation with sparse 3D points from stereo data", *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014.
- [5] J. Li et al., "A 3-D Real-Time Road Edge Detection System for Automated Smart Car Control", *2006 IEEE International Conference on Networking, Sensing and Control*, 2006.
- [6] K. Guo et al., "Road Edge Recognition Using the Stripe Hough Transform From Millimeter-Wave Radar Images", *IEEE Trans. Intell. Transp. Syst.*, pp. 1-9, 2014.
- [7] M. Nikolova and A. Hero, "Segmentation of a road from a vehicle-mounted radar and accuracy of the estimation", *Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No.00TH8511)*, 2000.
- [8] W. Zhang, "LIDAR-Based Road and Road-Edge Detection". *2010 IEEE Intelligent Vehicles Symposium*, pp 845-848, Jun. 2010.
- [9] B. Fardi et al., "Multi Model Detection and Parameter-based Tracking of Road Borders with a Laser Scanner", *IEEE IV2003 Intelligent Vehicles Symposium*, pp. 95-99, Jun. 2003.
- [10] W.S. Wijesoma, K. R. S. Kodagoda and A. P. Balasuriya, "Road-Boundary Detection and Tracking Using Ladar Sensing", *IEEE Trans. on Robot. Autom.*, vol. 20, no. 3, Jun. 2004.
- [11] G. Zhao and J. Yuan, "Curb detection and tracking using 3D-LIDAR scanner", *2012 19th IEEE International Conference on Image Processing*, 2012.
- [12] W. Yao, Z. Deng and L. Zhou, "Road curb detection using 3D LIDAR and integral laser points for intelligent vehicles", *The 6th International Conference on Soft Computing and Intelligent Systems, and The 13th International Symposium on Advanced Intelligence Systems*, 2012.
- [13] R. Li, et al., "LIDAR/MEMS IMU integrated navigation (SLAM) method for a small UAV in indoor environments", *2014 DGON Inertial Sensors and Systems (ISS)*, 2014.
- [14] J. Han et al., "Enhanced Road Boundary and Obstacle Detection Using a Downward-Looking LIDAR Sensor", *IEEE Trans. on Veh. Technol.*, vol. 61, no. 3, pp. 971-985, 2012.
- [15] S. Rodriguez et al., "Visual confirmation of mobile objects tracked by a multi-layer LIDAR", *13th International IEEE Conference on Intelligent Transportation Systems*, 2010.
- [16] P. Lindner et al., "Multi-channel LIDAR processing for lane detection and estimation", *2009 12th International IEEE Conference on Intelligent Transportation Systems*, 2009.
- [17] S. LaValle, *Planning algorithms*. New York, NY [u.a.]: Cambridge University Press, 2006.
- [18] Y. Liu and Y. Sun, "Mobile robot instant indoor map building and localization using 2D laser scanning data", *2012 International Conference on System Science and Engineering (ICSSE)*, 2012.
- [19] S. Jia et al., "Mobile Robot 3D Map Building Based on Laser Ranging and Stereovision," *Proceedings of the 2011 IEEE International Conference on Mechatronics and Automation*, 7 – 10 Aug. 2011.
- [20] M. Fu et al., "Multiple map representations for vehicle localization and scene reconstruction", *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014.
- [21] D. Cole and P. Newman, "Using laser range data for 3D SLAM in outdoor environments", *Proceedings 2006 IEEE International Conference on Robotics and Automation*, 2006.
- [22] S. Thrun et al., "Stanley: The robot that won the DARPA Grand Challenge", *J. Field Robotics*, vol. 23, no. 9, pp. 661-692, 2006.
- [23] M. Montemerlo et al., "Junior: The Stanford entry in the Urban Challenge", *J. Field Robotics*, vol. 25, no. 9, pp. 569-597, 2008.
- [24] S. Kuthirummal, A. Das and S. Samarasekera, "A graph traversal based algorithm for obstacle detection using LIDAR or stereo", *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.
- [25] H. Zhu et al., "A path planning algorithm based on fusing lane and obstacle map", *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014.
- [26] J. Gillula and J. Leibs, "How to teach a van to drive: an undergraduate perspective on the 2005 DARPA grand challenge", *IEEE Control Syst. Mag.*, vol. 26, no. 3, pp. 19-26, 2006.

CHAPTER 3

Implementation of Semantic Segmentation for Road and Lane Detection on an
Autonomous Ground Vehicle with LIDAR

Implementation of Semantic Segmentation for Road and Lane Detection on an Autonomous Ground Vehicle with LIDAR

Kai Li Lim, Thomas Drage and Thomas Bräunl

Abstract—While current implementations of LIDAR-based autonomous driving systems are capable of road following and obstacle avoidance, they are still unable to detect road lane markings, which is required for lane keeping during autonomous driving sequences. In this paper, we present an implementation of semantic image segmentation to enhance a LIDAR-based autonomous ground vehicle for road and lane marking detection, in addition to object perception and classification. To achieve this, we installed and calibrated a low-cost monocular camera onto a LIDAR-fitted Formula-SAE Electric car as our test bench. Tests were performed first on video recordings of local roads to verify the feasibility of semantic segmentation, and then on the Formula-SAE car with LIDAR readings. Results from semantic segmentation confirmed that the road areas in each video frame were properly segmented, and that road edges and lane markers can be classified. By combining this information with LIDAR measurements for road edges and obstacles, distance measurements for each segmented object can be obtained, thereby allowing the vehicle to be programmed to drive autonomously within the road lanes and away from road edges.

I. INTRODUCTION

The Renewable Energy Vehicle (REV) Project at the University of Western Australia conducts research into electric vehicles, vehicle automation and autonomous driving systems. Recent projects include the development of an Autonomous Formula-SAE Electric car [1]. This vehicle is an open-wheeled, electric drive race car, with electronic drive-by-wire and electromechanical brake/steering actuation. The vehicle serves as a compact, flexible test-bed for sensor testing and the development of autonomous driving algorithms.

Prior research has been conducted on road and road edge detection through optical systems [2], radar [3] as well as using Light Density and Ranging (LIDAR) sensors such as in the winning entry in the 2007 DARPA Urban Challenge [4]. The methodology described in [5] utilises a feature-extraction algorithm while other algorithms such as [6] rely on the presence of curbs and seek to identify and track curbs as features in the LIDAR data. More recently, there has been an increase in the use of cameras to achieve this [7], giving rise to visual road detection. Methodologies to achieve this include feature extraction and classification [7], horizon and vanishing point detections [8], and artificial neural networks (ANNs) [9].

The authors are with The REV Project at the School of Electrical, Electronic and Computer Engineering, The University of Western Australia, Perth WA 6009 Australia. {kaili.lim, thomas.braunl}@uwa.edu.au, thomas.drage@research.uwa.edu.au

The problem of path-finding can be described as: “Given a start state, a goal state, a representation of the robot and a representation of the world, find a collision-free path that connects the start with the goal satisfying the system constraints” [10]. In mobile robotics, a proven method to obtain the requisite representation of the world is via the use of LIDAR data to generate a virtual map in real-time both as the sole sensor [11] and in conjunction with data from additional sensors [12]. Similar LIDAR based map building approaches have been shown to be suitable for outdoor terrain [13]. These generated maps vary from simple two-dimensional maps suitable for basic path planning consisting of traversable regions, obstacles and unexplored regions [14] to more complex three-dimensional maps from which sophisticated cost maps are generated [15]. A more detailed map can be built by supplementing the camera in addition to LIDAR. These additional details can include a combination of vehicle detection and classification [16], road sign recognition [17], and scene recognition [18].

Visual cameras and LIDAR are often incorporated in autonomous driving systems. Works that combines LIDAR and camera sensors for autonomous driving include the approach from Zhang, Clarke and Knoll [16], where they have proposed the fusion of LIDAR and the camera as a compromise for each sensor’s drawbacks, with LIDAR providing range information, and the camera identifies objects and scenes. The authors achieved low false alarm rates and a high detection rate for vehicles in urban environments. A similar fusion of multiple LIDAR, radar, and camera sensors to achieve object detection and tracking was proposed by Cho et al. [19]. By tracking pedestrians and vehicles, the system could detect and track vehicles from 150m away, and pedestrians and cyclists within a 20 m radius. To the best of our knowledge, works that incorporate semantic segmentation onto a LIDAR-based autonomous ground vehicle has not been established at the time of writing.

Our work is an enhancement to the work done by Drage, Churack and Bräunl [20], where we have proposed a LIDAR-based road edge detection approach on the same vehicle. Our algorithm could detect road curbs and edges by measuring the differences in surface smoothness, which in turn allows the positioning of road edges and curbs.

II. IMPLEMENTATION

This section describes the addition of visual perception to the LIDAR-based autonomous SAE car as described in [20], which includes sections that describe our testing environment, and its applicable procedures to achieve visual

autonomous driving. By mounting a monocular camera onto the chassis of the vehicle, above the LIDAR (see Fig. 1), road recognition and obstacle detection are achieved using semantic segmentation. This camera supplements the LIDAR, where the LIDAR is responsible for providing distance measurements for objects and road edges detected by the camera. Semantic segmentation was achieved using SegNet [21], a convolutional neural network (CNN) architecture for semantic segmentation that is often used for road scenes. Its architecture uses an encoder-decoder network that is followed by a pixelwise classification layer, where the encoder and decoder networks consist 13 convolutional layers each. The Caffe [22] implementation of SegNet is used for this project. To interface the sensors for autonomous driving, SegNet is installed onto an Nvidia Jetson TX1 [23], and the LIDAR interfaces directly to a Raspberry Pi 3 [24], which drives a control system. A GPS module and an inertial measurement unit (IMU) module also connects to the Raspberry Pi 3 for positioning and localisation.

The LIDAR system used in this project consists of an IBEO Lux automotive LIDAR with specifications as shown in Table I. This sensor utilises reflected infra-red light to measure distance (via time-of-flight) and can build a 3D point cloud by scanning horizontally in four vertical layers. The IBEO sensor has sophisticated internal data processing functionality including object detection and classification. Data is delivered using TCP/IP over an Ethernet connection and includes scan data in polar coordinates and object data in $x-y$ coordinates referenced to the sensor.

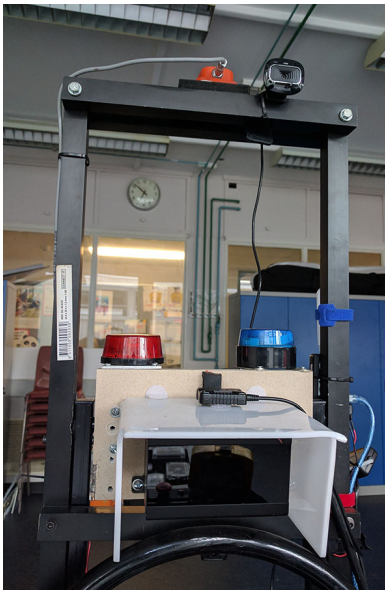


Fig. 1: The camera is mounted above the LIDAR system from [20], beside the IMU

The following subsections describe the process of achieving visual autonomous driving for our project using semantic segmentation with respect to its application environment and its driving sequences.

TABLE I: LIDAR Characteristics

Specification	Value
Technology	Time of flight (output of distance and echo pulse width)
Range	200m
Field of View (Horiz / Vert)	85 / 3.2
Layers	4
Echo Detection	3 measurements per pulse
Update Rate	Up to 50Hz
Accuracy	10cm

A. Application Environment

SegNet was tested within the grounds of the University of Western Australia (UWA), which is the same location that the LIDAR system was tested in [20]. The roads within UWA offers a similar drive environment to standard suburban roads. These single carriageway roads are of low traffic density, with views of pedestrians, faculty buildings, and vegetation for SegNet to recognise and segment. As a feasibility test, we also tested SegNet off a car-mounted dashcam recording while driving on local roads.

This application environment was selected to test the suitability of using SegNet for autonomous driving locally and to gauge the visual autonomous navigation performance of the vehicle. To achieve a successful autonomous drive using SegNet on the vehicle, road edges and lane markings must be properly recognised, before the application can be subsequently expanded onto a road-licensed vehicle.

It should be noted that our initial implementation uses the trained dataset from the University of Cambridge, CamVid [21]. This dataset was recorded in the City of Cambridge, England. Like most British cities, Cambridge's roads are often narrow, with dense buildings by the side. There is also a large pedestrian population due to it being an academic city. Comparatively, roads in Perth are generally wider, with a sparser build-up density than Cambridge. Its low population density means that there are fewer road pedestrians as compared to Cambridge. The ground terrain around Perth is mostly flat, with long sunshine hours. This means that one can generally expect excellent road visibility on Western Australian roads on most days. However, during poor visibility and night time drives, suburban roads around Perth are generally poorly lit, which may affect road segmentation accuracy. By testing this dataset, we will subsequently contemplate on the need to record and use a dataset from Perth for more accurate segmentation results.

B. Autonomous Driving Procedures

To use SegNet's output for autonomous driving, we assume that the car begins with a position in the middle of the road lane and that the road incline is flat. By mounting the camera at a fixed position on the car, the central driving position of the car can be obtained, along with distance measurements from the road edges to the left and right sides of the car. With the camera, this is done by identifying and segregating a fixed trapezoidal image region that

encapsulates the road segment, which is then transformed into a birds-eye view (BEV) perspective to obtain vehicle's position with respect to the road's centre. By scaling the road width according to the Australian standards of 3.3–3.5 m, along with the detected road edges and/or lane markings on SegNet's output, the distance from the vehicle's centre to the left and right road edges/lanes can be obtained as Fig 2, with its confidence value determined by the successful detection of road edges or lane markings. From these three sections distance thresholds for the left and right edges or lane markers distances for the car to autonomously centre itself on the road while driving. We call this road centring. In the event where lane markers are not found, road edges will be used instead.

To perform road centring, the car must steer itself in the opposite direction when it crosses the distance threshold to either the left or right road edge/lane markers. The distance from the car to the road edges or lane markers are constantly analysed. If the car is too close to the edge or lane marker, the road centring algorithm will then send commands to the drive system to steer away from the edges with fine adjustments, until the car is cleared from the distance threshold.

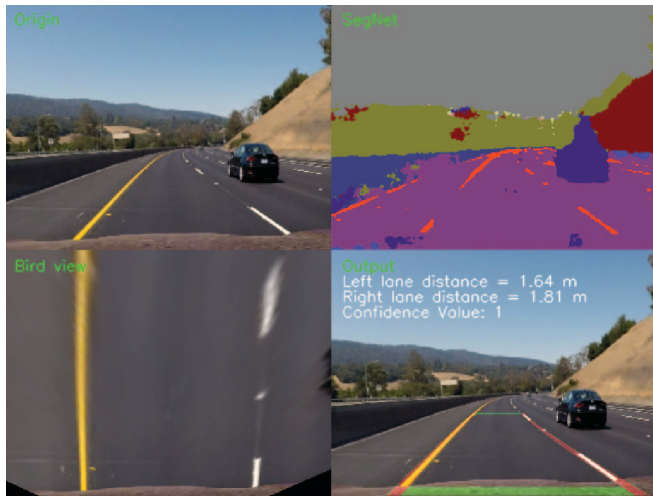


Fig. 2: Lane distance measurements with SegNet's output on Udacity's Self-Driving Car Nanodegree recording [25] as the input in *Origin*.

III. TESTING AND EVALUATIONS

A. Methodology

Testing begins with the calibration of the camera, where distance measurements in the real world will be represented in pixel ratios on SegNet's output. Here, we calibrated a Microsoft LifeCam HD-3000 camera. This was done by measuring the distances between road bollards in front of the parked vehicle on the road as illustrated in Fig. 3.

The bollards are placed a three-row formation to allow distance measurements from two-point distances on the frame. The first row of bollards on the car establishes the starting distance, with the centre bollard measuring the distance from the camera to the front of the car, while ensuring that the



Fig. 3: Photo illustrating the bollards' position with reference to the car in the centre.

camera is pointed to the centre of the car. All distances are measured from the base at the centre of each bollard. Subsequently, a topological representation of the measurements can be illustrated in Fig. 4, which is then represented again in the camera frame in Fig. 5. These measurements are verified with the LIDAR plot at that position as illustrated in Fig. 6, where the bollards (represented as dot plots) are clearly present around the 2 m, 8 m and 10 m mark on the y-axis. From Fig. 5, each image pixel was calculated to represent 17 mm and 23 mm when measured from 8.5 m and 11.6 m respectively from the camera, and that a level road will converge at around 41° on the camera frame.

The LIDAR readings complement SegNet's output for road edge detection, whereby we use our Kalman Filtered Linear Regression Model as described in [20]. Our algorithm minimises the square residuals between the fit line y and the data (x_i, y_i) , where the most suitable data line will be obtained for a given data set, and its success measured by the product-moment correlation coefficient r . The slope b and r values are tabulated as equation 1.

$$y = (\bar{y} - b\bar{x}) + bx, r = \frac{s_{xy}}{s_x s_y} \quad (1)$$

where:

$$b = \frac{s_{xy}}{s_x}, s_{xy} = \frac{\sum_{i=1}^n x_i y_i}{n} - \bar{x}\bar{y},$$

$$s_x^2 = \frac{\sum_{i=1}^n x_i^2}{n} - \bar{x}^2, s_y^2 = \frac{\sum_{i=1}^n y_i^2}{n} - \bar{y}^2$$

With a calibrated camera and LIDAR, the SAE car was driven around the application environment while the camera is recording. The recorded camera footage was used as an input for SegNet. For testing purposes, these footages were processed off-line with SegNet running off an Nvidia GTX Titan X GPU with a 480 by 360-pixel resolution, which resulted in a segmented image output at a consistent 10 frames per second (FPS) sampled at each video frame. This framerate is consistent with the visual autonomous driving results published by Nvidia in [26], making it adequate for autonomous driving. Further work is required for real-time on-line processing on the Jetson TX1. Likewise, LIDAR plots are also recorded during the duration of the drive, where timestamps are used to synchronise outputs.

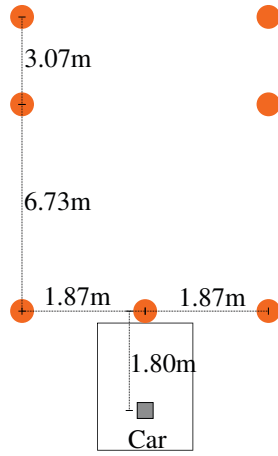


Fig. 4: The topological distance between bollards (dots) and the camera on the car (shaded square).

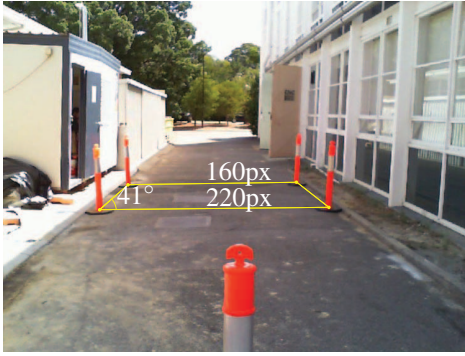


Fig. 5: Frame captured from the vehicle's camera for distance calibration

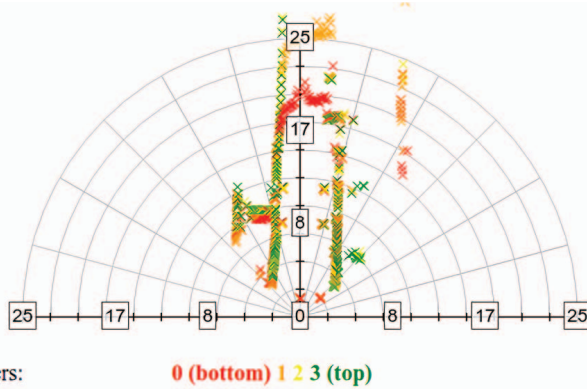


Fig. 6: LIDAR plot showing the detected road edges and the bollards' positions from the car where Fig. 5 was captured. The graph axes measure distances in metres.

B. Results and Discussions

Results from semantic segmentation are presented first on open roads with a dash cam recording, and then on the UWA campus ground with the SAE car. Image results in this section are presented with the left image showing segmentation input, and the right image showing its output through SegNet. In the case of false detections, they will

be measured in their pixel accuracy (PA). This is done by finding the percentages of road regions, which is done by counting the number of falsely detected pixels against the total number of pixels in the road region for that image frame.

Segmentation results from the open road testing from the dashcam footage yielded consistently favourable results, as shown in Fig. 7. Recordings were captured on a clear day, driving on a low traffic dual carriageway. The road region is accurately segmented with negligible false detections. In addition to the road and its markings, SegNet can detect and segment the speed limit sign and the right turn sign ahead. All vehicles and vegetation were also properly detected and segmented. The minor false detections in the sky region are due to the CamVid dataset being trained on a cloudy day, but this should not affect road detection accuracy in any way. This result confirms the feasibility that a CamVid-trained SegNet can be implemented for autonomous driving in the Perth metro area.



Fig. 7: SegNet's input (left) and output (right) for a dual carriageway in the Perth metro area.

Tests were subsequently performed on the SAE car for a vision-LIDAR-based implementation. Here, runs on campus grounds were recorded using the camera while manually driven on the SAE car following a predetermined route on campus. The car traversed across roads and pavements and the segmentation results are as follows.

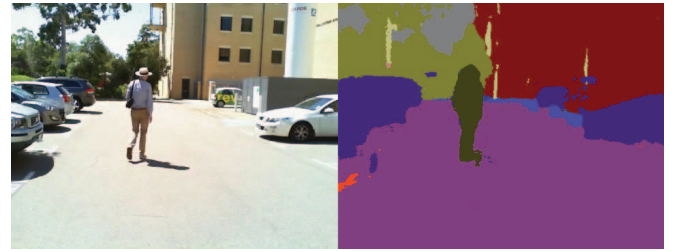


Fig. 8: Segmentation results from a parking area on campus grounds.

The test run began with a drive through the car park. SegNet's output in Fig. 8 shows that the image was segmented with good accuracy. With the exception of some minor (0.69%) false detections on the car's shadows on the left side, the road, parking lane, and pavements were properly segmented, along with the pedestrian and vehicles. With the LIDAR actively measuring the distance from the parked vehicles to the SAE car (see Fig. 9). Here, we adopt the

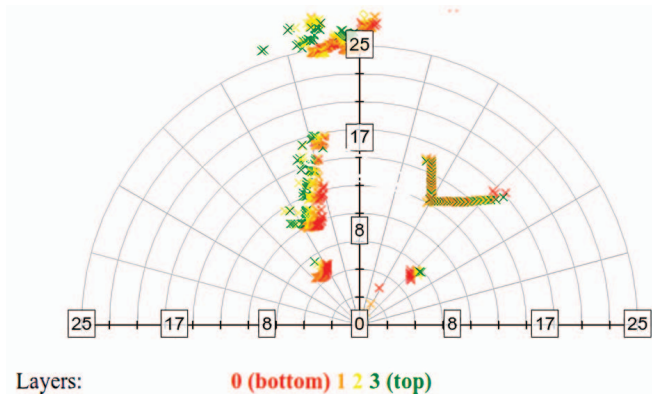


Fig. 9: LIDAR plot showing the detected parked vehicles at the position where Fig. 8 was captured. The graph axes measure distances in metres.

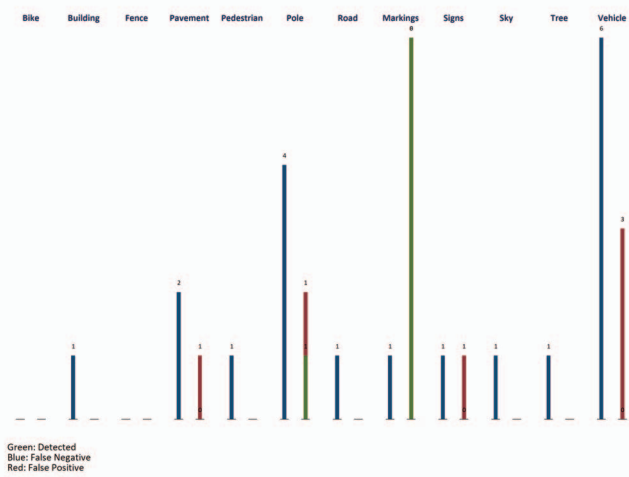


Fig. 10: The number of detected objects along with their errors in detections according to the objects.

Linear Regression model that we described in Section III-A, which plots the road edge position from the parked vehicles so that a fixed distance can be kept between the autonomous cars and the parked vehicles. We have also counted the number of detected objects along with their positive and negative false detections, which are plotted according to their detection/error pairs in Fig. 10, whereby the labelled number on each bar indicated the number of correctly identified objects, if present.



Fig. 11: Segmentation results on pavement between faculty buildings.

From the parking area, the car drives onto the pavement between faculty buildings. From Fig. 11, SegNet could discern pavements from roads as the grounds are now coloured blue. In addition, it was also able to detect the bicycles parked towards the right, and the pedestrians in the distance. False detections are present on the left side of the pavement, where it is incorrectly detected as buildings and pedestrians due to uneven lighting, accounting for 2.83% of the total pavement region.



Fig. 12: Segmentation results at a road junction.

Fig. 12 was captured when the car was stopping at a suburban road junction beside the campus. The segmentation output from this figure shows that while the objects in the distance were properly segmented, some parts at the bottom of the image were incorrectly classified as pavements and buildings, which makes up 16.19% of the road region. This was partly due to the clear weather resulting in a high brightness recording, while SegNet was expecting a darker surface to classify roads.



Fig. 13: Segmentation results on a road with pronounced shadows.

Fig. 13 was captured while the car was driving on a shadowed road on campus. The shadows on the road introduced a high contrast region, and that the bottom portion of the road was overexposed, resulting in a false classification of 13.93% and undetected lane markers. However, the overtaking vehicle, vegetation, road sign, and building were correctly segmented. When presented with a false detection on the road as with Figure 7, results from the LIDAR road edge detection system will compensate the regions of false detection, as the road and road edges were properly measured by the LIDAR system.

The false detection rates for each of the example figures is summarised as Table II, which also tabulates the number of road pixels in each detected road segments for each figure, along with the number of falsely classified pixels within that road segment. From these numbers, the detection error is

TABLE II: The number of road region pixels and its false detections pixels in that region for each figure along with their error percentages.

Fig.	Road Pixels	False Classifications	Error
7	41648	16	0.04%
8	86561	600	0.69%
11	76731	2168	2.83%
12	108684	17595	16.19%
13	83831	11679	13.93%

calculated as a percentage that corresponds to the area of each figure's road segments. The false detection rate is at its highest in Fig. 12, where the road region encompasses most of the frame, the unevenness in road surface and lighting is the main contributor to this error rate. Conversely, Fig. 7 records the lowest false detection rate its road segment, as the road area was well-defined, and the frame was properly exposed.

IV. CONCLUSION

We have presented a semantic segmentation-based visual navigation approach for autonomous ground vehicles. This approach improves on existing LIDAR-based vehicles to introduce object recognition and classification while driving. SegNet adequately performs semantic segmentation to recognise roads and lane markers, which in turn allows the vehicle to maintain a safe distance from the road and lane edges in addition to LIDAR measurements. We have also shown that the segmentation results from SegNet on the CamVid dataset is satisfactory on Perth metro roads. With a calibrated camera, visual autonomous driving can be achieved using real-time semantic segmentation. Future works will focus on the complete on-line implementation of SegNet on the SAE car for real-time visual autonomous driving.

ACKNOWLEDGMENT

The authors would like to thank The REV Project's Autonomous SAE team for their efforts in aiding the compilation of data for this paper, and Nvidia Corporation for sponsoring the GPU under its GPU Grant Program.

REFERENCES

- [1] T. Drage, J. Kalinowski, and T. Braunl, "Integration of drive-by-wire with navigation control for a driverless electric race car," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 4, pp. 23–33, 2014.
- [2] P. Y. Shinzato, D. Gomes, and D. F. Wolf, "Road estimation with sparse 3d points from stereo data," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, Conference Proceedings, pp. 1688–1693.
- [3] K. Y. Guo, E. G. Hoare, D. Jasteh, X. Q. Sheng, and M. Gashinova, "Road edge recognition using the stripe hough transform from millimeter-wave radar images," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 825–833, 2015.
- [4] W. Zhang, "Lidar-based road and road-edge detection," in *2010 IEEE Intelligent Vehicles Symposium*, 2010, Conference Proceedings, pp. 845–848.
- [5] M. Nikolova and A. Hero, "Segmentation of a road from a vehicle-mounted radar and accuracy of the estimation," in *Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (Cat. No.00TH8511)*, 2000, Conference Proceedings, pp. 284–289.
- [6] G. Zhao and J. Yuan, "Curb detection and tracking using 3d-lidar scanner," in *2012 19th IEEE International Conference on Image Processing*, 2012, Conference Proceedings, pp. 437–440.
- [7] Y. Alkhorshid, K. Aryafar, S. Bauer, and G. Wanielik, "Road detection through supervised classification," in *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2016, Conference Proceedings, pp. 806–809.
- [8] P. Y. Shinzato, V. Grassi, F. S. Osorio, and D. F. Wolf, "Fast visual road recognition and horizon detection using multiple artificial neural networks," in *Intelligent Vehicles Symposium (IV)*, 2012 IEEE, 2012, Conference Proceedings, pp. 1090–1095.
- [9] N. Abbas and V. Mahdi, *A Novel Neural Network based Voting Approach for Road Detection via Image Entropy and Color Filtering*, ser. Indian Journal of Science and Technology.
- [10] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.
- [11] Y. Liu and Y. Sun, "Mobile robot instant indoor map building and localization using 2d laser scanning data," in *2012 International Conference on System Science and Engineering (ICSSE)*, 2012, Conference Proceedings, pp. 339–344.
- [12] F. Mengyin, Z. Hao, Y. Yi, W. Meiling, and L. Yu, "Multiple map representations for vehicle localization and scene reconstruction," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, Conference Proceedings, pp. 2241–2242.
- [13] D. M. Cole and P. M. Newman, "Using laser range data for 3d slam in outdoor environments," in *Proceedings 2006 IEEE International Conference on Robotics and Automation*, 2006. ICRA 2006., 2006, Conference Proceedings, pp. 1556–1563.
- [14] Z. Hao, F. Mengyin, Y. Yi, W. Xinyu, and W. Meiling, "A path planning algorithm based on fusing lane and obstacle map," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, Conference Proceedings, pp. 1442–1448.
- [15] J. Gillula and J. Leibs, "How to teach a van to drive: an undergraduate perspective on the 2005 darpa grand challenge," *IEEE Control Systems*, vol. 26, no. 3, pp. 19–26, 2006.
- [16] F. Zhang, D. Clarke, and A. Knoll, "Vehicle detection based on lidar and camera fusion," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, Conference Proceedings, pp. 1620–1625.
- [17] A. Gudigar, S. Chokkadi, and R. U, "A review on automatic detection and recognition of traffic sign," *Multimedia Tools and Applications*, vol. 75, no. 1, pp. 333–364, 2016. [Online]. Available: <http://dx.doi.org/10.1007/s11042-014-2293-7>
- [18] J. Yang, S. Zhang, G. Wang, and M. Li, "Scene and place recognition using a hierarchical latent topic model," *Neurocomputing*, vol. 148, pp. 578–586, 2015.
- [19] H. Cho, Y. W. Seo, B. V. K. V. Kumar, and R. R. Rajkumar, "A multi-sensor fusion system for moving object detection and tracking in urban driving environments," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, Conference Proceedings, pp. 1836–1843.
- [20] T. Drage, T. Churack, and T. Braunl, "Lidar road edge detection by heuristic evaluation of many linear regressions," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 2015, Conference Proceedings, pp. 2465–2470.
- [21] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *arXiv preprint arXiv:1511.00561*, 2015.
- [22] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, Conference Proceedings, pp. 675–678.
- [23] NVIDIA Corporation, "Embedded systems," 2017. [Online]. Available: <http://www.nvidia.com/object/embedded-systems-dev-kits-modules.html>
- [24] Raspberry Pi, "Raspberry pi 3 model b," 2017. [Online]. Available: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>
- [25] Udacity, "Carnd-lanelines-p1," 2017. [Online]. Available: <https://github.com/udacity/CarND-LaneLines-P1>
- [26] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, and J. Zhang, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

CHAPTER 4

A Modular Software Framework for Autonomous Vehicles

A Modular Software Framework for Autonomous Vehicles

Kai Li Lim, Thomas Drage, Roman Podolski, Gabriel Meyer-Lee,
Samuel Evans-Thompson, Jason Yao-Tsu Lin, Geoffrey Channon, Mitchell Poole and Thomas Bräunl

Abstract—Software frameworks for autonomous vehicles are required to interface and process data from several different sensors on board the vehicle, in addition to performing navigational processes such as path planning and lane keeping. These can include a combination of cameras, LIDARs, GPS, IMU, and odometric sensors to achieve positioning and localisation for the vehicle and can be challenging to integrate. In this paper, we present a unified software framework that combines sensor and navigational processing for autonomous driving. Our framework is modular and scalable whereby the use of protocol buffers enables segregating each sensor and navigation subroutine individual classes, which can then be independently modified or tested. It is redesigned to replace the existing software on our Formula SAE vehicle, which we use for testing autonomous driving. Our testing results verify the suitability of our framework to be used for fully autonomous drives.

I. INTRODUCTION

UWA's Renewable Energy Vehicle Project (REV) operates two autonomous vehicles, a BMW X5 and a student-built Formula SAE-Electric car. Formula SAE [1] is a long-running annual competition organised by the Society of Automotive Engineers with competition events worldwide. The design competition includes petrol cars, as well as the SAE-Electric class which includes ours with two electric motors driving each of the two rear wheels via independent controllers. We have incorporated full drive-by-wire control of the throttle, steering and the hydraulic braking system. The use of a Formula-SAE car provides several advantages for such a project; the car is mechanically simple, Formula-SAE cars with similar designs are common at universities worldwide and the size of the car makes testing accessible without forgoing race car vehicle dynamics. Furthermore, the use of an electric vehicle makes the project significantly more practical for student work as the risks and environmental issues associated with petrol-engine cars are eliminated and the systems installed in this project can take advantage of the large amount of electrical energy already available on the vehicle.

For the driverless FSAE project, our goal was to be able to autonomously drive a vehicle around a race track. This is being achieved by placing waypoints along the ideal driving line, as well as "fence points" to lock out non-driving areas. Maps can either be recorded by human or remote-controlled

driving or specified through a Google Maps driven web-interface. During autonomous driving, a laser scanner and camera are used for detection of road edges as well as any obstacles on the track. Safety systems are essential for a driverless system, as the car weighs more than 250 kg and is capable of driving at a speed of 80 km/h. Both the low-level drive-by-wire, as well as high-level navigation system have independent safety systems built in. These include remote intervention, remote heartbeat and remote emergency stopping, which are implemented through a fail-safe wireless link to a base station as well as through hard-wired buttons on the car itself.

Our motivation for designing and presenting this framework is to improve upon the existing autonomous drive software on our SAE vehicle that is documented in [2]. This software utilises a web-based user interface (UI) that displays via a touch screen mounted in the vehicles cockpit. Our proposed framework utilises this existing UI with a revamped backend as described in Section II. It was noted that the existing software had a heavy reliance on a central Control module, which required all the sensors and their submodules to run to function. These submodules were programmed throughout the years with different programming languages and redundancies, which made integration difficult. Our proposed framework presents a streamlined approach whereby each module will be programmed with a C++ interface that communicates with either a path planner or a drive control system. This system also benefits from additional features including visualisation data playback (online or offline) and a web-based user interface. As a customised framework for our project, this also alleviates the reliance on node-based solutions such as ROS, which usually requires a perpetually running Master node to function and allows higher reliability when the individual components are integrated.

Additionally, this approach presents a long-term advantage whereby our framework is made fully open and contributable by students and enthusiasts looking to implement our framework onto their custom-fabricated vehicles. When compared against other autonomous driving frameworks such as Apollo [3] and Autoware [4] that mostly target commercial vehicles and requiring expensive hardware, our approach leverages on hardware and fabrication methods that are more accessible in cost.

II. AUTONOMOUS DRIVING FRAMEWORK

This section introduces our proposed software framework for the SAE vehicle, with the software architecture diagram as illustrated in Fig. 1. The software architecture of the

¹The authors are with The REV Project at the School of Electrical, Electronic and Computer Engineering, The University of Western Australia, Perth WA 6009 Australia. {kaili.lim, thomas.braunl}@uwa.edu.au, thomas.drage@research.uwa.edu.au, roman.podolski@tum.de, gmeyer11@swarthmore.edu, {20490234, 21680206, 21317528, 21212271}@student.uwa.edu.au

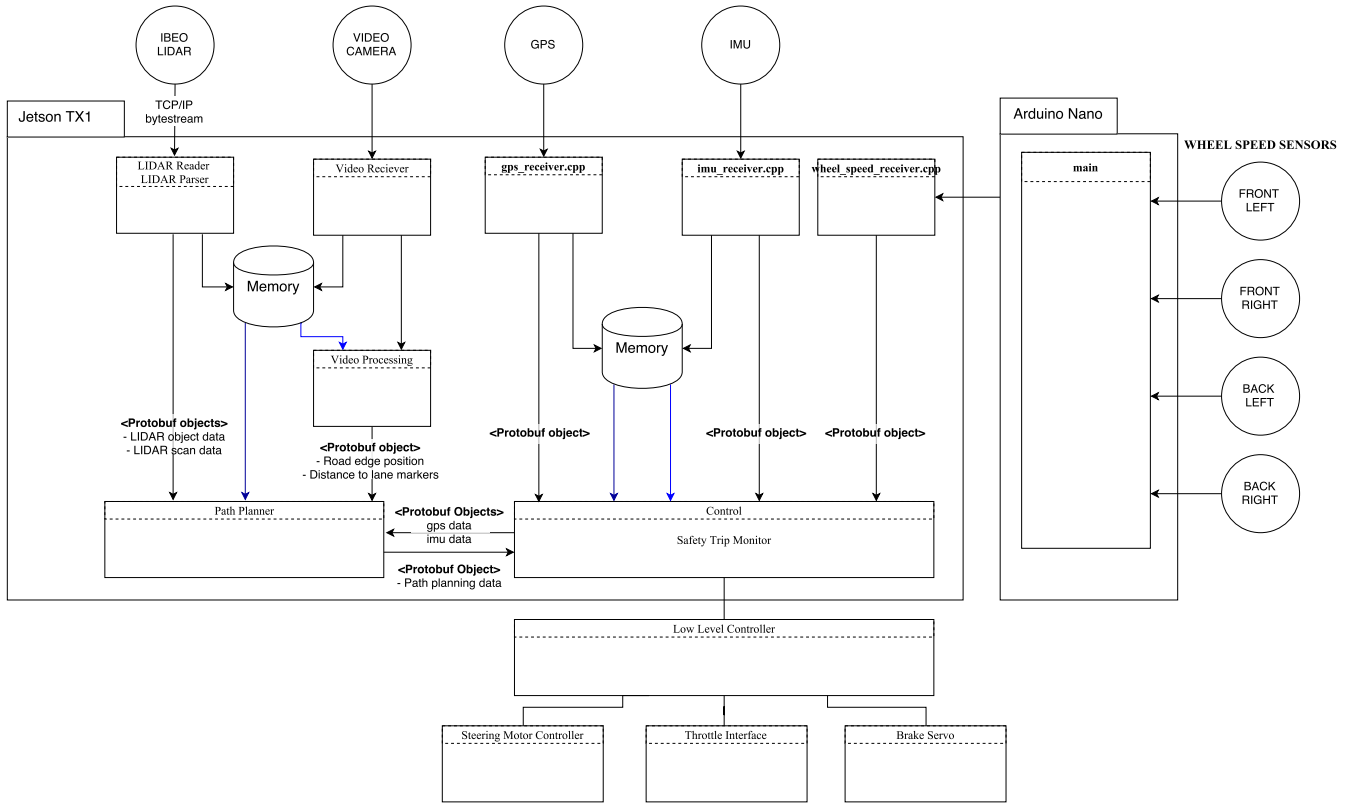


Fig. 1. The software architecture diagram of our proposed framework

vehicle utilises a modular design to allow for continuous parallel development on each of the sensors, the path planning algorithm, and the control policy. Input sensors are comprised of the LIDAR, camera, GPS, IMU and four wheel speed sensors, which are required to function simultaneously in order for the car to drive autonomously. The LIDAR, camera, GPS and IMU each have their own receiver class on the Jetson TX1, which takes the input from the sensors and processes the data. Additionally, rotary-encoder odometry is performed using an Arduino microcontroller connected to each of the wheels' encoder. This information is received in the high-level software by another receiver class which processes the data to produce wheel speed measurements. Relevant data for path planning and localisation is encoded in protobuf [5] format and then passed to either the path-planning program or the control program, which performs localisation as well as some communication and logging utilities. The control program also communicates with the web GUI, providing a visualisation of the data and allows the user to start and stop autonomous driving as well as the safety trip monitor and the controller, thereby introducing a high-level interface for driving the car. The control program, once it has communicated with the path-planner to combine localisation data with a set of future way points, will transmit driving instructions to the controller. This in turn transmits them to a low-level microcontroller. Not pictured in the diagram is the fusion of data from sources such as the GPS and IMU, and IMU and LIDAR in order to improve the accuracy of localisation. Detailed explanations of the sensors and classes are covered in Sections II-A to II-H.

A. Path Planner

We use a real-time capable path-planning algorithm based on [6]. Given a set of pre-recorded or pre-defined waypoints along a track, the planner will generate a optimised path through all way points, which serves as a baseframe for trajectory generation. During operation, the algorithm evaluates a variety of possible trajectories in the configurations space of the vehicle using RRTs [7]. Those intermediate trajectories are generated along a curvilinear coordinate system, along with the baseframe. Each possible trajectory is checked for collisions with obstacles. A collision free path is then picked, utilizing a cost-function, that enables us to influence the driving style of the vehicle. The algorithms are implemented in C++14 and rely heavily on the C++ source libraries Boost and Eigen3.

We use an arc-length parametrised cubic B-spline $P_b(s)$ [8] to generate a baseframe for the curvilinear coordinate system, which can be described as a non-linear transformation of the parameter domain on the four waypoints a to d , parametrised by s .

$$P_b(s) = \begin{cases} x_b(s) &= a_{x,i}(s - s_i)^3 + b_{x,i}(s - s_i)^2 \\ &+ c_{x,i}(s - s_i) + d_{x,i} \\ y_b(s) &= a_{y,i}(s - s_i)^3 + b_{y,i}(s - s_i)^2 \\ &+ c_{y,i}(s - s_i) + d_{y,i} \end{cases} \quad (1)$$

The curvature κ_b can be calculated from the first and second derivatives of $P_b(s)$

$$\frac{dx_b}{ds} = x'_b = \cos \theta_b \quad \frac{dy_b}{ds} = y'_b = \sin \theta_b \quad (2)$$

$$\kappa_b = \frac{x'_b y''_b - x''_b y'_b}{(x'^2_b - y'^2_b)^{\frac{3}{2}}} \quad (3)$$

The curvature of a cubic spine is continuous in all sections. For this reason, a parametric cubic B-spline is adopted for the baseframe. Since the position of the vehicle is provided in Cartesian-space, we need to find a transform those coordinates to a curvilinear representation, of which the B-spline provides the base. This is equal to finding the closest point to the vehicle on the baseframe, by minimising the Euclidean distance between the position of the vehicle and the cubic B-spline.

$$\min_s d(s) = \sqrt{(x_v - x_b(s))^2 + (y_v - y_b(s))^2} \quad (4)$$

We choose Brent's method [9] find the minimum. Another suitable and stable algorithm is provided in [10]. From the lateral distance to the baseframe and the baseframe we construct the curvilinear coordinate system in which we generate a number n of paths as cubic polynomials. Each polynomial is defined by a lateral offset $q(s)$ and a curvature κ , to cover a broad section of the configuration space of the vehicle. n is a design parameter and can be chosen to adjust the computational load of the algorithms. We now design a vehicle model of a set of differential equations in Cartesian space.

$$\dot{x} = v \cos \theta, \dot{y} = v \sin \theta, \dot{\theta} = v \kappa \quad (5)$$

This vehicle model is transformed onto the curvilinear coordinate system, and the position of the vehicle in Cartesian space can then be determined by forward integration. Paths that violate the non-holonomic constraints of vehicle motions or collide with an obstacle are eliminated. The remaining paths are evaluated by a construction. The cost function itself can be chosen to optimise driving for an arbitrary property, like sportiness or safety. By simulating the path planner using equally weighted costs, the near-optimal path can be obtained as Fig. 2(a), with the path drawn in green and the baseframe in blue. The manoeuvre selection is subsequently presented as Fig. 2(b). Because this planner insoles a simple vehicle model as a set of ordinary differential equations, it can be conveniently integrated into any control algorithm.

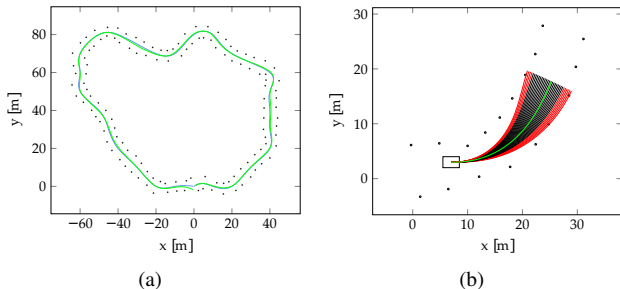


Fig. 2. Simulated near-optimal path (a) and manoeuvre selection (b). Black dots represent road delimiters.

B. Software Communications

The sensors on the vehicle communicate with the path planner using protocol buffers. Protocol Buffers (protobuf) [5] are a formalised data structure developed by Google for use in cross-platform systems. Protobuf allows for the combination of several standard variables into a single packed structure that can be easily serialised and accessed using automatically generated methods. The protobuf library has bindings for many common programming languages, including C++, Python and Java, meaning that a protobuf structure packaged in a C++ application, can be read in by a Java application with no conversion needed.

Protocol buffers are used internally within our software as regularly structured state variables, with easy to use member functions. For example, the GPS software stores its current state within a protobuf object containing data such as latitude, longitude, groundspeed and angle. This GPS state can then be used by internal code regularly, or it can be used serialised and stored. The serialised protobuf data is very compact and space efficient, meaning that a huge amount of logged data can be saved sequentially.

By abstracting the actual data from a specific sensor behind a protobuf object, it allows for the use of Polymorphism within our software, and so dependencies on a specific piece of hardware are loosened. As long as a specific hardware device can be interpreted into the appropriate protobuf form, it can be integrated easily into the system, with only short wrapping code needed to be written. There is also no dependency for this protobuf data to come from a physical sensor, the protobuf data can be read in from a previously serialised log, allowing for the replay of data, or it can be read from an external piece of software, allowing for the use of simulation programs.

C. Localisation

The vehicle achieves localisation through the inertial measurement unit (IMU) and global positioning system (GPS). The IMU used in the project is the Xsens MTi [11] 9 depth-of-field sensor. The sensor contains several advanced internal algorithms in order to provide accurate and noise-free measurements of the current gyroscopic position, the acceleration, and the magnetic field. These values are returned by the IMU readings as the velocity, acceleration, and the three rotations (pitch, roll and yaw) along the x , y and z axis. The sensor is used within the project to determine heading, and assist in the calculation of local positions.

The GPS receiver used is the Columbus V-800 GPS receiver. It is used with the GPSD [12] software to parse the National Marine Electronics Association (NMEA) standard data outputted by the GPS unit, and retransmit internally in an easier to use format. The data used from the GPS unit are the GPS coordinates, the ground speed, and the heading. These GPS coordinates are first converted into a local reference frame, as a distance from a datum point. The acceleration and position data from the IMU and the GPS units respectively are fused together using an extended Kalman filter (EKF), producing a value for positioning that

has a higher accuracy than GPS alone. This fusion system outputs the filtered position, velocity, and acceleration data which is then fed into the Path Planner and Control modules along with the bearing to ensure that the vehicle can reliably localise itself and obtain accurate readings for the position, velocity and acceleration.

D. Odometry

The SAE car has been fitted with Hall Effect sensors that send its data through a comparator and an OR gate, this makes a pulse train where we use an Arduino UNO [13] to count the time between pulses to give angular velocity, which can be translated to meters per second. This gives the car Odometry, in which software will use an EKF to fuse the measurements together to improve their individual measurements and the vehicles localization capabilities. As the Arduino UNO is too slow to control the steering as well as breaking for the SAE Car, we added a second Arduino to do the odometry which then sends the data through serial communications to the main Arduino Nano. This low-level communicates steering and wheel velocity to the Nvidia Jetson TX1 for processing the data through the EK Filter, using a simple car kinematic model. The goal is to achieve the localization with reduced reliance on the GPS.

E. LIDAR

The Autonomous Formula SAE vehicle uses an ibeo LUX [14] Light Distance and Ranging system (LIDAR) to sense distance information about the world around it. The LIDAR records the time interval between emission and recapture of thousands of infra-red light pulses to record a stream of 3-Dimensional points. The LIDAR is specifically designed for automotive purposes and is capable of internal data analysis; detecting and classifying objects in its field of vision. The LIDAR connects via an Ethernet switch to the Nvidia Jetson TX1 [15]. A LIDAR reader class receives the serial bytestream which is continuously being transmitted. The data parsing is handled by the parser class which converts the bytestream into Protobuf objects. This format facilitates the storage and sharing of the information to the LIDAR visualisation.

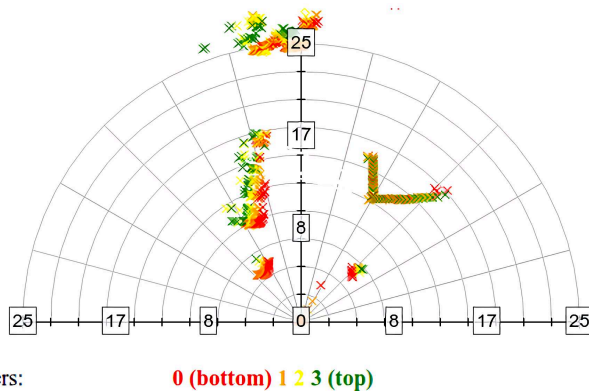


Fig. 3. LIDAR plot showing the detected parked vehicles at the position where Fig. 4 was captured. The graph axes measure distances in metres.

Road-edge detection is achieved through the use of the LIDAR data. The LIDAR, aimed at an angle below horizontal beyond the front of the car, provides four layers of depth information with a horizontal angle of 85 degrees. Its output is visualised as Fig. 3. Road edge detection is achieved by analysing the depth information in one of the layers and checking it for both smoothness and slope. The central data points and those near them are considered and checked to confirm that they meet the slope criteria (the road should be relatively flat so no great changes in depth should be noted in a line) iteratively further and further points are considered in a stepwise process where the correlation coefficient is considered at each point. The road edge is the point at which the correlation coefficient is the highest whilst the slope condition is still being met. This approach was improved with the implementation of a Kalman filter which creates a time-averaged estimate of the road edge-position assisting in the prediction of the current road edge.

F. Vision

The SAE vehicle uses visual information as one of its references for driving decisions. It mainly uses an off-the-shelf monocular camera to collect images then applied through OpenCV and SegNet [16] for road edges detection. OpenCV provides many modules, such as image processing, video, and video I/O, that is useful for road edges detection. However, using OpenCV alone for image recognition is limited by variations in image quality, brightness, and contrast. SegNet is an image semantic segmentation approach. It can classify road scene objects, such as the pedestrian, lane marking, traffic light, vehicles etc., that complement the insufficient of single image processing scheme. SegNet is a pixel-wise semantic segmentation in deep learning framework. Semantic segmentation is used to apply for understanding the visual scene and object. This has been widely adopted in autonomous driving. The architecture of SegNet is a convolution encoder and decoder which is a pixel-wise classifier. The objects classify from SegNet is including following classes, sky, Building, column-pole, road-marking, road, pavement, tree, sign-symbol, fence, vehicle, pedestrian, and bicyclist. The accuracy of classify result is 65.9% for classes average [16]. The input images utilise SegNet to perform visual scene classification. This will produce results whereby road, road-marking, and pavement are classified (see Fig. 4), which is useful for road edges detection. OpenCV is simultaneously used to perform image processing. The first step for image processing is camera calibration to get an undistorted image. This is achieved using a chess board image and finding chess board corners to get two accumulated list – 3D point in real world space and 2D points in an image plane. Then we use the camera calibration function in the OpenCV library to obtain the camera calibration and distortion coefficients. This scheme will remove camera distortions.

The road edges detection scheme detects lane-marking at two sides of autonomous SAE vehicle. The lane-marking detection can also be performed solely by the OpenCV library. Finding the edges of the whole image will reduce the

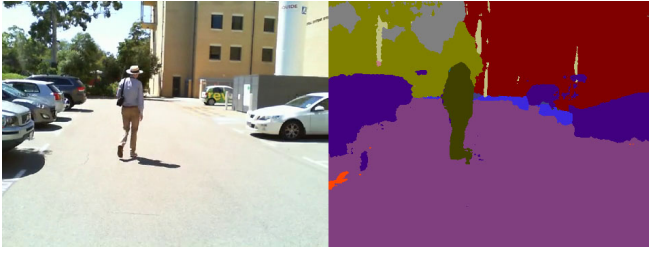


Fig. 4. Segmentation results from a parking area on campus grounds.

image complexity because numbers of colour and gradient of the image would make image processing more difficult. Canny edge detection [17] is a convenient approach in found in the OpenCV library that can be applied for this purpose. Then, Hough transform [18] can be applied onto the image to detect the lanes on both sides of autonomous SAE vehicle. The marking of lanes is detected then using perspective transforms to get a birds eye view-like image. It can easily find the four points to represent the lane marking pair, where a second order polynomial method can be applied to fit the points. The lane distances are obtained using pixel values that are converted into metres. The scaling factors are according to Australia road width standard 3.3 to 3.5 meters. However, the image processing approach might fail because the lane markings are not clear or no lane markings. Therefore, using SegNet's results can effectively circumvent this drawback due to its ability to robustly detect and classify road and lane markings, whereby the same road distance calculation can be applied to find the vehicle's distances to the road edges.

G. Safety Trip Monitor

The safety trip monitor was designed with an observer-notifier structure. Any of the objects responsible for performing a safety-crucial function in the software can call a trip on the trip state monitor. This includes the low-level safety software, the controller, the GPS software, the web interface, the heart beat and the car network. The trip state is stored by the trip state monitor and pushed to any object which implements the trip state observer class and has registered itself with the monitor. The observer class ensures that the trip state does not produce any irregular operations while driving. The observers which receive the trip state upon each change are attached to the monitor after its instantiation, which means that the set of objects in the software which can change the trip state and which need to track to trip state can be completely reconfigured without needing to make changes to the trip state monitor or observer classes. This is an ideal structure for the software of a research autonomous vehicle, as the continuous development of ongoing research will frequently modify the structure of functions of portions of the software while seeking to maintain the integrity of safety features, like the safety trip.

H. Controller

The controller class is the high-level interface for the drive-by-wire functionality of the vehicle. The actual drive-by-wire controlling of the vehicle is done by separate soft-

ware on an Arduino micro controller. The controller class is the only high-level software which communicates with the low-level controller. The program utilizes three PID controllers to set the throttle, brake, and steering values. The controller class provides a high-level interface with methods to begin to stop autonomous control of the throttle, brake, and steering, as well as methods to set the bearing or speed of the vehicle with desired values. This interface is utilized by the Control program, which handles path planning, and the Fusion program, which provides fused IMU-GPS data in order to facilitate waypoint-based driving. Fig. 5(a) and 5(b) illustrates the baseframe and curvature output by the path planner using the Control program based on our evaluation path in Section III. A large change in curvature is present where the U-turn was made.

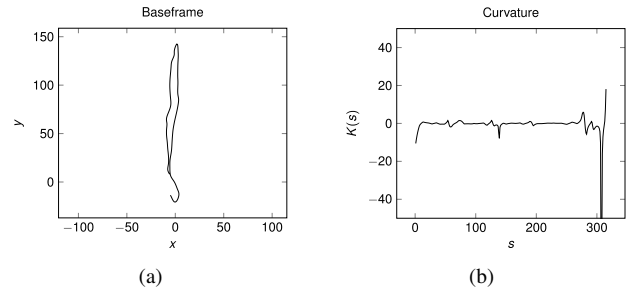


Fig. 5. The trajectory baseframe generated by the path planner (a) and its derived curvature (b).

III. IMPLEMENTATION ON AN SAE VEHICLE

The software for the autonomous driving system is programmed onto an Nvidia Jetson TX1 embedded computer that is mounted on the chassis of the vehicle. The environmental sensors namely the LIDAR, GPS, IMU and camera connects directly to the camera via Ethernet (LIDAR) and USB 2.0 (GPS, IMU, camera) respectively. This software can be implemented onto another vehicle so long as the same sensors are used, as the system outputs drive commands through the Control module, which can be configured according to the vehicle's hardware specifications. To test our system, we collected driving data with the vehicle driving in a parking area at the University of Western Australia (shown in Fig. 6) by recording readings from the GPU, IMU, LIDAR and camera. The GPS and IMU plots waypoints for the car, LIDAR performs obstacle detection, and the camera performs semantic segmentation. The test drive begins from the southern end and then driving towards the northern end before making a U-turn back to the vehicle's station position. The recorded waypoints are passed to the path planner, which generates a trajectory baseframe as shown in Fig. 5(a). Subsequently, the curvature is obtained from the derivative of the baseframe as Fig. 5(b), which can then be used to determine the steering angle for autonomous driving.

IV. RESULTS

Fig. 4 was captured while the car was driving northward as it approaches the end of its path. The input image is displayed on the left and its semantic segmentation result is displayed on the right image. Results from semantic



Fig. 6. Map showing the path taken by the vehicle in with a solid red line. segmentation showed that the road is properly classified, along other elements in the frame. Its LIDAR readings at that position is as illustrated in Fig. 3, whereby the parked vehicles are detected on the left, along with the vegetation in the distance and the wall on the right side of the vehicle. The combination of LIDAR and semantic segmentation enables the vehicle to understand its position on the road, along with the obstacle types and the distances to each obstacle. For further results on road and lane marker detection, we processed a drive recording from Udacity's Self-Driving Car Nanodegree [19]. From Fig. 7, the input image (Origin) is used for both semantic segmentation (SegNet), and bird's eye view (BEV) transformation (Bird view). The system was able to identify objects on the road scene, and the curvature of the road can be calculated using the BEV. The distance between the centre of the vehicle and the left and right lane markers are calculated as shown in the output. This is accompanied with a confidence value whereby a successful detection of the road lane markings will be denoted with a '1'.



Fig. 7. Original and Jetson TX1 output showing segmentation, birds eye view, vehicle distance to left and right lane marks.

V. CONCLUSION

In this paper, we have designed and demonstrated the functionality of our software framework that is implemented on our autonomous SAE vehicle. This framework is designed to cohesively interface with the vehicle's camera, LIDAR, GPS, IMU and wheel speed sensors while being capable

of performing navigational tasks such as path planning, image processing, odometry, localisation, safety checks, speed and steering control. Each sensor and navigation task is programmed as a separate module to ensure modularity and scalability, allowing for each module to be changed independently. Protocol buffers handle intermodular communications, whereby each process parses its output as a protobuf to be sent to another module. With this framework implemented on the Jetson TX1, our test drives on the autonomous SAE vehicle was able to achieve results that are adequate for fully autonomous driving. Future works will include further testing of the autonomous navigation software and refinements to the control and path planner classes to ensure that the system is capable for road drives using full automation.

ACKNOWLEDGMENT

The authors thank Nvidia Corporation for providing the GPU and the Jetson TX1 under its GPU Grant Program.








REFERENCES

- [1] SAE International, "Student Events - Events - Collegiate Design Series." [Online]. Available: <https://www.sae.org/attend/student-events/>
- [2] T. H. Drage, "Development of a Navigation Control System for an Autonomous Formula SAE-Electric Race Car," Master's thesis, The University of Western Australia, 2013. [Online]. Available: <http://robotics.ee.uwa.edu.au/theses/2013-REV-Navigation-Drage.pdf>
- [3] Baidu, "Apollo." [Online]. Available: <http://apollo.auto/>
- [4] Autoware, "Autoware." [Online]. Available: <https://autoware.ai/>
- [5] Google Developers, "Protocol Buffers." [Online]. Available: <https://developers.google.com/protocol-buffers/>
- [6] K. Chu, M. Lee, and M. Sunwoo, "Local path planning for off-road autonomous driving with avoidance of static obstacles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1599–1616, Dec 2012.
- [7] S. M. Lavalle, "Planning Algorithms," *Journal of Chemical Information and Modeling*, vol. 53, no. 9, pp. 1689–1699, 2013.
- [8] H. Wang, J. Kearney, and K. Atkinson, "Arc-length parameterized spline curves for real-time simulation," in *Proc. 5th International Conference on Curves and Surfaces*, 2002, pp. 387–396.
- [9] R. P. Brent, *Algorithms for Minimization without Derivatives*. Englewood Cliffs, N.J.: Prentice-Hall, 1973.
- [10] J. Xu, W. Liu, H. Bian, and L. Li, "Accurate and efficient algorithm for the closest point on a parametric curve," in *2008 International Conference on Computer Science and Software Engineering*, vol. 2, Dec 2008, pp. 1000–1002.
- [11] Xsens, "MTi (legacy product) - Products." [Online]. Available: <https://www.xsens.com/products/mti/>
- [12] E. S. Raymond, "GPSd Put your GPS on the net!" [Online]. Available: <http://www.catb.org/gpsd/>
- [13] Arduino, "Arduino Uno Rev3." [Online]. Available: <https://store.arduino.cc/usa/arduino-uno-rev3>
- [14] AutonomouStuff, "ibeo Standard Four Layer Multi-Echo LUX Sensor | LiDAR | Product." [Online]. Available: <https://autonomoustuff.com/product/ibeo-lux-standard/>
- [15] NVIDIA Corporation, "Embedded systems," 2017. [Online]. Available: <http://www.nvidia.com/object/embedded-systems-dev-kits-modules.html>
- [16] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *arXiv preprint arXiv:1511.00561*, 2015.
- [17] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, Nov 1986.
- [18] P. Hough, "Method and means for recognizing complex patterns," Dec 1962.
- [19] Udacity, "Carnd-lanelines-p1," 2017. [Online]. Available: <https://github.com/udacity/CarND-LaneLines-P1>

CHAPTER 5

Evolution of a Reliable and Extensible High-Level Control System for an
Autonomous Car

Evolution of a Reliable and Extensible High-Level Control System for an Autonomous Car

Kai Li Lim , Thomas Drage , Chao Zhang , Craig Brogle , William W. L. Lai , Timothy Kelliher ,
Manuchekhr Adina-Zada, and Thomas Bräunl 

Abstract—The reliability of autonomous vehicles is heavily dependent on their software frameworks, which are required to interface and process data from many different sensors on board the vehicle, perform navigational processes such as path planning and lane keeping, take action to ensure safety and display data to an operator in a useful fashion. These sensors can include a combination of cameras, LiDARs, GPS, IMU, and odometric sensors to achieve positioning and localisation for the vehicle and nearby objects in their environment and can be challenging to integrate. In this paper, we present a hybridised software framework that combines sensor and navigational processing for autonomous driving. Our framework utilises a modular approach for interfacing and safety functionality, whilst navigation and sensor interfaces are implemented as nodes in the robot operating system. Our testing results verify the suitability of our framework by integration with a hardware-in-the-loop simulation system and for fully autonomous field driving.

Index Terms—Autonomous driving, software framework, sensor fusion, driving simulation.

I. INTRODUCTION

THE Renewable Energy Vehicle Project (REV) at UWA has developed an intelligent autonomous test vehicle, utilising a Formula-SAE race car (see Fig. 1) as a platform (Formula SAE [1] is an annual student competition organised by the Society of Automotive Engineers with events worldwide). Using such a vehicle allows us to target driving applications, both on- and off-road, in structured and unstructured driving environments. We have incorporated full drive-by-wire control of the electric vehicle's throttle, steering and the hydraulic braking system. The use of a Formula-SAE car provides several advantages for such a project as the vehicle is mechanically simple. Formula-SAE cars with similar designs are common at universities worldwide and the size of the vehicle makes testing accessible. Furthermore, the use of an electric vehicle makes the project



Fig. 1. Autonomous SAE Vehicle.

significantly more practical for student work and the hardware installed in this project can take advantage of the large amount of electrical energy already available on the vehicle.

For the driverless FSAE project, our goal was to be able to autonomously drive a vehicle around a race track. Initially, this was achieved by placing waypoints along the ideal driving line, as well as “fence points” to lock out non-driving areas. Maps can either be recorded by human or remote-controlled driving or specified through a Google Maps driven web interface. During autonomous driving, a laser scanner and camera are used for detection of road edges as well as any obstacles on the track. Our current work involves increasing the level of automation to drive a semi-structured (traffic cone or road edge delineated) race track by first automatically driving and mapping the path without prior knowledge of the track and then re-driving it, optimised, at greater speed with the assistance of inertial navigation.

Safety systems are essential for a driverless system, as the car weighs more than 250 kg and is capable of driving at a speed of 80 km/h. Both the low-level drive-by-wire, as well as high-level navigation system have independent safety systems built in. These include active geofencing, remote intervention, remote heartbeat and remote emergency stopping, which are implemented through a fail-safe wireless link to a base station as well as through hard-wired buttons on the vehicle itself.

The previous revision of our framework [2], [3] had a heavy reliance on a central control module, which required all the sensors and their submodules to function. These submodules were developed over time using different programming languages and are partially redundant, which made integration difficult. First,

Manuscript received September 15, 2018; revised February 27, 2019; accepted May 15, 2019. Date of publication June 3, 2019; date of current version August 23, 2019. (Corresponding author: Kai Li Lim.)

The authors are with the The REV Project, the School of Electrical, Electronic and Computer Engineering, The University of Western Australia, Perth, WA 6009, Australia (e-mail: kaili.lim@uwa.edu.au; thomas.drage@research.uwa.edu.au; 21435393@student.uwa.edu.au; 21313578@student.uwa.edu.au; 21696421@student.uwa.edu.au; 21305667@student.uwa.edu.au; 21135495@student.uwa.edu.au; thomas.braunl@uwa.edu.au).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TIV.2019.2919459

this was streamlined in an approach whereby each module is programmed with a C++ interface that communicates with either a path planner or a drive control system. Here, we present the integration of this approach with the Robot Operating System [4] which allows the further separation of functions into ROS nodes and provides a consistent application programming interface (API) for implementation of additional sensors and higher level automation, whilst the original program, now running as a ROS node, deals with critical interfacing and safety functions.

Additionally, this approach presents a long-term advantage whereby our framework is made fully open and contributable by students and enthusiasts looking to implement our framework onto their custom-built vehicles. When compared against other autonomous driving frameworks such as Apollo [5] and Autoware [6] that mostly target commercial vehicles requiring expensive hardware, our approach leverages hardware and fabrication methods that are more affordable. The framework has been integrated with a simulator which provides the ability to test software modifications and algorithms prior to deployment to the vehicle.

Our contributions in this paper are demonstrated through the proposal of our high-level autonomous control system that interfaces through standard, off-the-shelf sensors and equipment. This system is made holistic through the integration of the following features — real-time localisation through odometry and dead-reckoning; object segmentation and detection using LiDAR and camera; real-time path planning via waypoints or object positioning; visual navigation with odometry, object recognition and tracking, and semantic segmentation; and a hardware-in-the-loop simulator for prototyping verification.

The remainder of this paper is organised as follows. Section II introduces the system framework with an overview. Section III describes the sensors that are used. Section IV presents the path planning approaches used in the system. Section V explains how visual navigation is performed on the system. Section VI highlights our driving simulator. Section VII describe our system validations, before the concluding remarks are drawn in Section VIII.

II. SYSTEM OVERVIEW

In order to satisfy the requirements of resiliency, flexibility, extensibility and simple integrations, a publish/subscribe software architecture was used. This allowed for highly decoupled software to be developed, with each component or series of components needing only to conform to the expected message type on the input and output topics. The use of a publish/subscribe architecture allows for nodes to be easily swapped in and out in order to test different solutions, as well as allowing multiple components to make use of the same data sources without modifying the source, providing simple methods of logging, data capture and data replay.

It was decided to use ROS as the publish/subscribe layer of the application. This decision was influenced heavily by the usage of ROS in the Apollo Auto [5] project, as it is shown to be reliable and performant. This also provides a path for upgrades, with a version of ROS modified for automotive use through the addition of shared memory transport for message passing, support for the

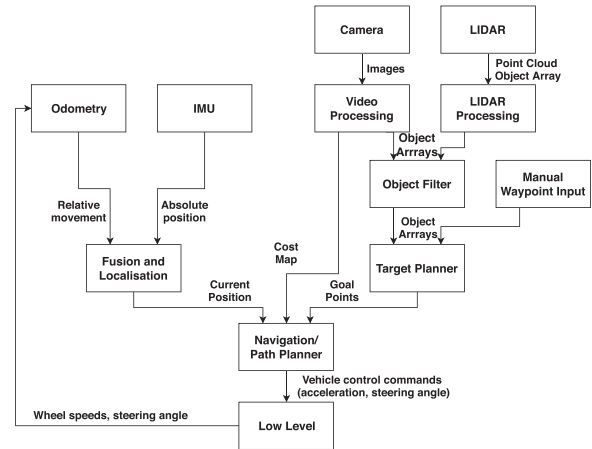


Fig. 2. SAE vehicle software framework.

Protobuf [7] message passing protocol, and the decentralisation of ROS to reduce single points of failure available on an open-source licence. This also allows for any components developed to be more easily ported to the complete Apollo Auto platform at a later stage. In addition, the popularity of ROS ensures that there are a large number of existing modules available for use, allowing the team to focus on the goals stated above, instead of on creating supporting code and utilities. Based on the desired functionality, it was determined that the publish/subscribe architecture would initially require the nodes and topics outline in Fig. 2.

We migrated the existing code base of the high-level software system to use the ROS framework in 2018 to reduce the development complexity of the software system. ROS provides low-level device control, implementation of commonly used tools, message passing between processes, and package management [4]. Instead of creating an independent system where a *broker* would manage the intercommunications between *modules* (programs that have a specific function) ROS readily provides these services. Hence, the user only has to create *nodes* (programs that perform a certain function) that listen and talk to other nodes. Using ROS therefore simplifies the integration process for each individual module in the system. By defining the topic information for messages to communicate, the individual nodes can work together without too much effort in integration.

More specifically, existing modules in our system presented in [3], including modules for logging, web server and serial communication were replaced with their equivalent ROS packages, as they are often more stable and better supported. All existing messages from the system are converted into ROS messages. The testing for the individual modules therefore only requires minimal changes to the core software.

The ROS version used on the SAE vehicle is currently ROS Kinetic Kame running on Ubuntu 16.04 LTS which will be long-term supported until April 2021.

III. NAVIGATION SENSORS

The SAE vehicle performs autonomous driving through a combination of navigation sensors including LiDARs, cameras,

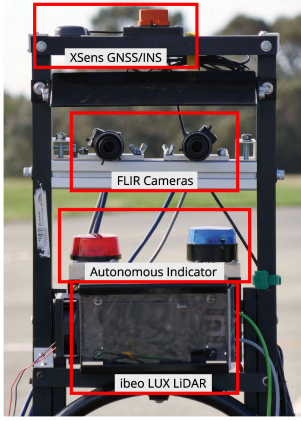


Fig. 3. The SAE vehicle's rack-mounted sensors.

wheel odometry and inertial measurement unit (IMU) (as shown in Fig. 1, with the sensor rack detailed in Fig. 3) which are categorised into four categories — odometry, dead reckoning, LiDAR and camera systems, which are elaborated upon in their individual Sections III-A to III-D.

A. Odometry

The SAE vehicle has been fitted with Hall Effect sensors on each wheel which send data through a comparator and an OR gate, and generate a pulse train to an Arduino Nano [8]. The sensors count pulses for each wheel and report them to the low-level controller with timestamps. The linear velocity and rotational velocity are then evaluated by a low-level controller with the feedback from the steering sensor. The accumulated pose information is then combined with the wheel odometry of the SAE car. The goal for implementing this wheel odometry is to provide basic offline localisation within a low-level system and to be further fused with other sensors such as IMU, LiDAR and cameras for a more accurate global positioning.

B. Dead Reckoning

Having an accurate state estimation is crucial for making optimal decisions for future control inputs to effectively navigate the environment. Dead reckoning on the vehicle is achieved through the Xsens MTi-G-710 [9], which is an inertial measurement unit (IMU) that is equipped with a global navigation satellite system (GNSS) running at 50 Hz. However, these sensors are susceptible to noise and imperfections which introduce uncertainty to the measurements. Hence, we introduce an extended Kalman filter (EKF) to fuse data from these sensors with that from odometry using a model of the car's dynamics to obtain a more precise estimate of its state. Since the computation of the car's movements requires direction, trigonometric functions are needed. The EKF linearises these non-linear functions using a first-order Taylor series approximation [10], where it is approximated according to [11] in equation (1).

$$f(u_k, x_{k-1}) \approx f(u_k, \mu_{k-1}) + \frac{df(u_k, \mu_{k-1})}{dx_{k-1}}(x_{k-1} - \mu_{k-1}) \quad (1)$$

where u and μ are the mean and the estimate of x , respectively. With an EKF, we calculate the fused covariance values P_k by predicting next state and the next error covariance using the current state and current estimate error covariance. x_k is the current pose at time instance k and f is a non-linear transition function that converts the past state to the current state; state x is composed of the car's x - y coordinates and orientation φ . We perform this as a three-step process — 1. Compute the Kalman gain K , 2. Perform the correction to find the current state x_k , and 3. Calculate the new process error P_k ; following the descriptions in Section III of [12]. This is used as the foundation for our experiments on sensor fusion, as described in Section VII-A.

C. LiDAR System

The vehicle utilises an array of Light Distance and Ranging (LiDAR) systems. This consists of a SICK LMS111-1010 [13] and an ibeo LUX 4 [14] connected through an Ethernet switch to the Nvidia Jetson TX1. The LMS111 scans a single layer at 50 Hz while the LUX scans four layers at 10 Hz featuring in-built object detection and tracking. The data is published by each of the LiDAR's ROS drivers and processed to achieve desired functionalities.

The LMS111 is mounted forward-facing on the front of the vehicle at 15 cm above the ground to obtain a 270° field of view, suitable for detecting close obstacles and scans a plane close to horizontal. The point cloud is sorted and then from left to right; each point is assigned a cluster identification number based on distance to the next point and the angle between it and the next point relative to the laser. This delivers accurate obstacle detection with features such as cluster size indicating the size of the obstacle, allowing for classification of obstacles (such as a cone). The LUX is mounted above the driver and pitched towards the ground slightly such that the lower layer scans the ground 20 metres ahead of the vehicle. It is utilised to achieve road edge and object detection at a distance.

Road edge detection is achieved by analysing the depth information in one of the layers and checking it for both smoothness and slope. The central data points and those near them are considered and checked to confirm that they meet the slope criteria (the road should be relatively flat so no great changes in depth should be noted in a line). Iteratively, further and further points are considered in a stepwise process where the correlation coefficient is considered at each point. The road edge is the point at which the correlation coefficient is the highest whilst the slope condition is still being met. This approach was improved with the implementation of a Kalman filter which creates a time-averaged estimate of the road edge-position assisting in the prediction of the current road edge. A detailed description of our methods is presented in [15].

The in-built object detection and tracking of the LUX will be used for fusion with the obstacles reported through processing of the LMS111's data. The comparison of positions of objects reported by the LUX and LMS111 will increase the likelihood of detecting an object. The LMS111 giving information on the objects on a low and horizontal plane and the LUX giving information of objects in the mid to far range. In turn, this object information will also be fused with that of the camera vision.

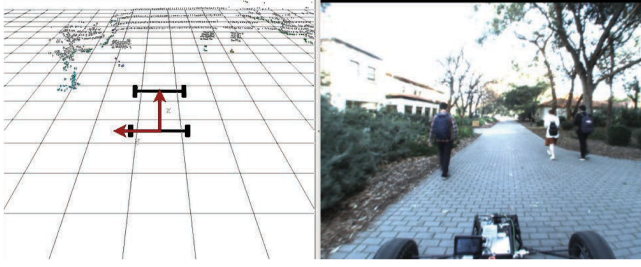


Fig. 4. Point cloud generated from the LMS111 (coloured) and the LUX (4-layers, white) (left) and the scene where it was captured (right).

We use both LiDARs to collectively generate a point cloud as illustrated in Fig. 4, captured during a test run. The LMS111 point cloud is coloured based on its intensity, retrieving information on the reflectivity of the surface struck by each point. The LUX is scanning layers onto the path in the distance while also picking up a large amount of detail from the surrounding vegetation as pedestrians walk past.

D. Camera System

A pair of FLIR Blackfly GigE [16] cameras are mounted on the vehicle's frame to perform tasks related to visual navigation, such as semantic segmentation and visual odometry. These cameras are fitted with Fujinon f/1.2, 2.8–8 mm wide-aperture varifocal lenses [17], and are individually capable of capturing a wide field of view. To suit our application, these cameras use 1.3 MP 1/3" global shutter CCD image sensors that will not be affected by any distortions caused by the rolling shutter effect [18]. The cameras are connected to a Gigabit Ethernet switch that connects to the Jetson TX1, interfacing them through Blackfly's ROS node where it is configured to record at 10 Hz per channel.

At the time of writing, the use of stereoscopic vision for autonomous driving is still subject to evaluation. Our application focuses on using monocular vision that is paired with measurements from the LiDAR system in order to achieve depth perceptions. The methods that we use for semantic segmentation and visual odometry for this paper currently do not require a stereoscopic setup. Our implementation of these methods is further described in Section V.

IV. PATH PLANNING

We have programmed the control system to deliver path planning routines to drive either through a series of predefined waypoints, or in between a series of traffic cones placed on either side of the vehicle.

A. Waypoint Driving

The underlying idea behind waypoint driving is to drive a set of predefined points in between the starting position P_1 ($x = 0$, $y = 0$ and orientation $\varphi = 0$) to the destination position P_2 , which can be obtained through the differences in GPS coordinates. These waypoints can either be stored in Cartesian coordinates in an array or in our test case, they are selected based on the driver's preference by selecting position points on RViz [19], which are

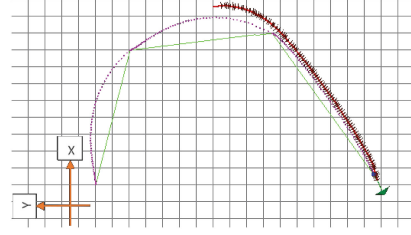


Fig. 5. Calculated waypoints on RViz with inputs from equations (2) and (3).

then confirmed on the console. To ensure that all points can be driven smoothly with the consideration of the correct heading to the subsequent point, a spline approach has been implemented within this design to generate the desired path.

Once the waypoints are chosen, two static paths are shown on RViz (see Fig. 5). The first path (green) consist of straight lines that interlink all the points with the arrow at the end showing the destination heading. The second path (purple) is the desired path which consists of a smoothed curvature that passes through all of the waypoints, the generation of this path is based on the Hermite spline interpolation technique [20] with the required parameters which are the four vectors — current position P_1 , target position P_2 , tangent of departure from current position T_1 and tangent of approach from target position, T_2 ; along with four Hermite basis functions $H_n(u)$

$$\begin{aligned} H_1(u) &= 2u^3 - 3u^2 + 1 & H_2(u) &= -2u^3 + 3u^2 \\ H_3(u) &= u^3 - 2u^2 + u & H_4(u) &= u^3 - u^2 \end{aligned} \quad (2)$$

where $0 \leq u \leq 1$ which represents the start to finish motion. We then construct the resultant path f by calculating the product between the vectors and the Hermite basis functions

$$f(x, y, \varphi) = H_1P_1 + H_2P_2 + H_3P_3 + H_4P_4 \quad (3)$$

The actual simulated driving pattern (red line) is determined based on the distance measurement between the current position of the vehicle against the desired path (purple) with a predefined tolerance range, and limited maximum turning angle ranged between $\pm 25^\circ$. The desired path is constituted by a finite number of points generated from the Hermite spline interpolation. To avoid oversteering or understeering, we compute the slope differences for both the driving path and the desired path. With this logic in place, the vehicle is either turning left ($\varphi > 0^\circ$), right ($\varphi < 0^\circ$) or moving straight either at a constant speed or slowed speed for a sharper turn in order to reach the goal point ($\varphi = 0^\circ$).

B. Cone Driving

The current iteration of the path planning procedure uses obstacle detection of the cones to determine the correct path. This algorithm is similar to our solution in [3], but simplified to allow for quicker calculation. Our cone driving module accepts cone locations from either the map, LiDAR or camera, classifying them as objects. Then, the vehicle navigates to drive within the track formed by cones safely without collision. Using a range of the maximum turning circle of the car, of both a left-hand turn

Algorithm 1: Cone Driving.

```

1: procedure CONEDRIVE (cones in range)
2:   init steering_range to  $[-25, 25]$ 
3:   for all cones in range do
4:     evaluate collision_range with cone
5:     exclude the collision_range from
       steering_range
6:   end for
7:   if steering_range is empty then
8:     stop
9:   else if all steering_range  $\leq$  threshold then
10:    select largest steering_range
11:   else if all steering_range  $>$  threshold then
12:    select steering_angle with minimum change in
       current direction
13:   end if
14:   drive toward centre of steering_range
15: end procedure

```

and right-hand turn, it then looks at which predicted paths will intercept cones. The vehicle dynamics are thus limited during motion planning such that the steering angle does not exceed 25° . Our algorithm will iterate through all cones within the car's range and calculate the best collision-free path to undertake, as detailed in Algorithm 1.

V. VISUAL NAVIGATION

The vehicle is capable of performing visual navigation tasks through a combination of semantic segmentation, visual odometry and visual cone detection that is decided depending on the application requirements. These tasks commonly rely on the OpenCV [21] library, utilising functions such as handcrafted feature detection, general image processing and transforming. It achieves a visual perception of the driving environment through the camera system as described in Section III-D.

A. Road and Lane Detection

Our system uses either semantic segmentation or edge detection to detect road edges and lane markings, depending on the environment's complexity. Using semantic segmentation also enables obstacle recognition which can be integrated with the LiDAR system. Environments are complex when there is a lack of uniformity in pose, features and illumination. While it is possible to solely rely on modules within OpenCV here, the performance of using handcrafted features alone for image recognition is restricted by variations in image quality, brightness, and contrast. In order to improve its performance under these environments, we selected SegNet [22] for semantic segmentation due to its high compatibility and ease of implementation. Its ability to perform pixel-wise classification for road scene objects complements the drawbacks of a single image processing scheme.

The architecture of SegNet uses a convolution encoder and decoder setup that classifies objects into one of the following classes — sky, building, column-pole, road-marking, road,

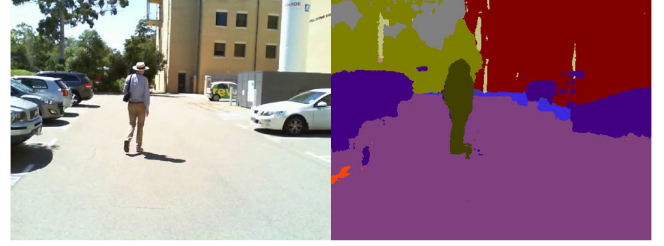


Fig. 6. Semantic segmentation results during a test drive. This scene resulted in a pixel accuracy of 99.31%.

pavement, tree, sign-symbol, fence, vehicle, pedestrian and bicyclist; with a class average classification accuracy of 65.9% [22]. Our application uses SegNet whereby road, road markings, and pavement are classified (see Fig. 6), which is useful for road edges detection. However, OpenCV is simultaneously used to perform image processing, with the first step being camera calibration to get an undistorted image. This is achieved using a chessboard image and finding its corners to get two accumulated list — a 3D point in real-world space and a 2D point in an image plane. We then use the camera calibration function in the OpenCV library to obtain the camera calibration and distortion coefficients. Our experiments using SegNet for autonomous driving was performed in [23], where we have evaluated its segmentation accuracy in our test environment through the calculation of its pixel accuracy (PA).

$$PA = \frac{\sum_i n_{ii}}{\sum_i t_i} \quad (4)$$

where n_{ii} represents the number of classified class i pixels correctly classified to belong in class i , and t_i represents the total number of pixels in class i belonging in the ground truth.

The road edges detection process finds lane markings at both sides of the car. We have noted that lane marking detection can also be performed solely using OpenCV functions, especially in non-complex, uniform environments with minimal illumination variations, thereby reducing its computation requirements. Algorithm 2 describes our approach.

The lane distances are obtained using pixel values that are converted into metres, and its scaling factors are according to Australian road width standard of 3.3 to 3.5 metres. However, this image processing approach might fail when the lane markings are not clear or there are no lane markings. Therefore, using SegNet's results can effectively mitigate this drawback due to its ability to robustly detect and classify road and lane markings, whereby the same road distance calculation can be applied to find the vehicle's distance to the road edges.

B. Visual Odometry

Our system implements ORB-SLAM2 [24] as our baseline algorithm for visual odometry based on its use of Oriented FAST and rotated BRIEF (ORB) features for mapping, tracking and place recognition. ORB features are similar to Binary Robust Independent Elementary Features (BRIEF) with an extra feature of introducing rotation invariance and noise resistance, while utilising Features from accelerated segment test (FAST) for

Algorithm 2: Road and Lane Detection.

```

1: procedure LANEDETECT (histogram_vector from
   camera)
2:   undistort image_frame from lens distortions
3:   Sobel filter image_frame as filtered_image
4:   threshold filtered_image as binary_image
5:   obtain histogram_vector for binary_image on
      y-axis
6:   split histogram_vector into left_half and right_half
7:   for each histogram_vector do
8:     find peak_position on y-axis
9:     init sliding_window at bottom of image at
       peak_position
10:    while sliding_window not at top_of_image do
11:      find mass_center of sliding_window as
        line_point
12:      move window to the mass_center
13:      move window iteratively on x-axis towards
        top_of_image
14:    end while
15:    fit line_point with second-order
      polynomial
16:  end for
17: end procedure
    
```

corner detection. This results in a balanced compromise between accuracy and computation footprint, making it desirable for our hardware setup. Although initially proposed as a visual simultaneous localisation and mapping (SLAM) algorithm, our ORB-SLAM2 application focuses on visual odometry as we do not implement its loop closing feature. To further increase the efficiency of this algorithm for our specific needs, a new set of vocabularies were trained using the images collected from the cameras mounted on the car. This reduces the size of vocabularies, which results in improved memory footprint. The Jetson TX1's 256-core embedded GPU is being used to improve image processing through parallelisation. The original ORB-SLAM2 was not adapted for this acceleration. In order to boost runtime performance, the applied ORB-SLAM2 has been rewritten to adapt CUDA [25] and thus utilise the GPU on the TX1 [26]. Fig. 7 illustrates an experiment with ORB-SLAM2 on a path segment as shown in (a), with its generated path in (b) through the tracking of features along subsequent frames.

C. Cone Detection

Handcrafted features that combine a linear classifier are utilised for cone detection using OpenCV. We use the histogram of oriented gradients (HOG) descriptor across an image to find and segregate regions of interest (ROIs) that may encompass a cone, which is then used as inputs for a support vector machine (SVM) classifier. For all regions that are positively classified, the hue layer is thresholded with an orange value, as our system is benchmarked using orange cones. We finally apply a histogram to the thresholded image and then obtain the position of the cone within the image frame. However, in order to fuse this classification result with other sensors, the detected cones

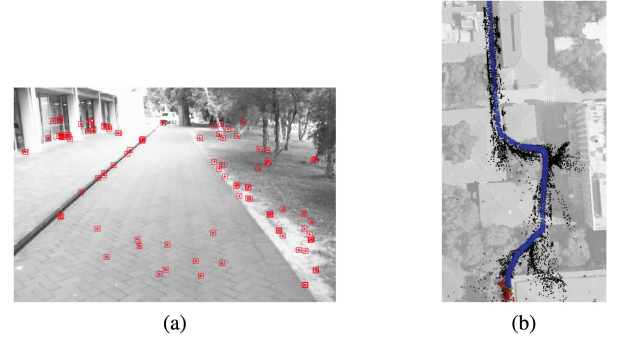


Fig. 7. ORB-SLAM2 experiment showing (a) tracked keyframes as red points that results in (b) the generated path in blue with the said tracked features as red dots, overlaid on satellite imagery.

must be presented in the global reference frame. This is done by applying a perspective transform to the image, and with the assumption that the vehicle is driving on a flat plane. The position of the cones in the global frame can then be obtained by projecting these cones onto a horizontal ground plane.

In order to reduce the effect from variations in brightness caused by the different sunlight angles, more samples must be included in the training process of the SVM. This significantly degrades the runtime performance of the entire system while offering only a minor accuracy improvement. Because of this, we designed a convolutional neural network (CNN) to provide a flexible feature extraction method to adapt this variation in the environment. This network has two convolutional layers and two fully-connected layers, which are used for detecting cones. In order to run the visual cone detection process smoothly along with all other modules in the system, we designed the network to be simple with a small footprint. Using this CNN yielded increased detection accuracies as compared to the SVM approach while offering similar runtime performance [27].

VI. HARDWARE-IN-THE-LOOP SIMULATION

Simulation is a cornerstone of autonomous vehicle testing, allowing high-level software such as image processing and path planning to be tested in predefined scenarios on a much faster schedule than is possible with hardware testing. In addition, the use of autonomous driving simulations allows for testing to be performed in faster than real time, and for testing to be scheduled and the results reviewed at a later time.

We designed a hardware-in-the-loop (HIL) simulation system, whereby an interface between the high-level software of the autonomous SAE vehicle and the CARLA driving simulator [28] allows software components such as localisation and path planning to be tested in a simulated environment, resulting in faster iterations as tests can be conducted at any time that is convenient. This interface consists of a ROS node that retrieves environment data such as a camera feed and LiDAR point cloud from CARLA and sends movement commands to CARLA. Due to the message passing system used by ROS, this interface can easily generate and consume the outputs and inputs expected by other ROS nodes without requiring changes to the application, even across multiple devices, and allows for the application architecture in Fig. 8. This allows CARLA to be run on a

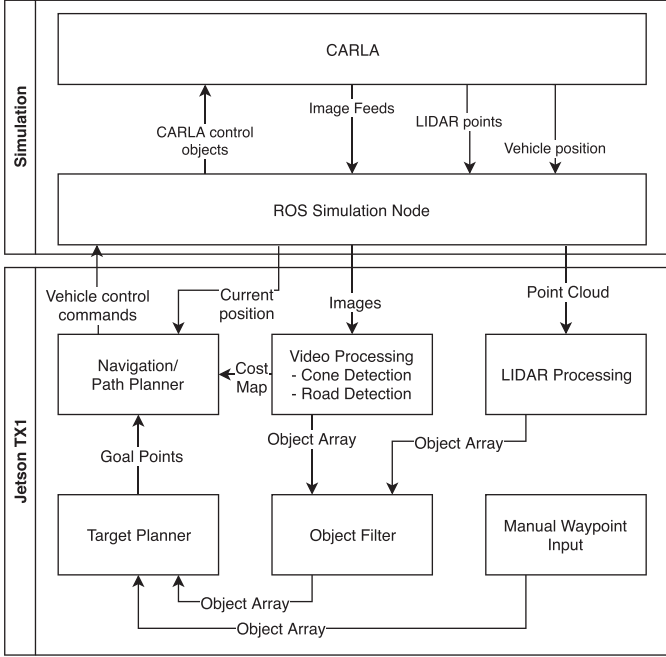


Fig. 8. Simulation software framework.

TABLE I
DISPLACEMENT & ERROR COVARIANCE COMPARISON

IMU Fusion	k	Δx_k	P_k
No	50.88	(1.430, 0.030)	(0.0160, 0.0160)
	100.91	(41.070, 13.934)	(0.0160, 0.0160)
	150.89	(58.000, 27.680)	(0.0160, 0.0160)
Yes	50.88	(1.436, 0.025)	(0.0130, 0.0134)
	100.91	(41.073, 13.901)	(0.0132, 0.0132)
	150.89	(57.970, 27.658)	(0.0132, 0.0133)

more powerful computer more suited to generating a simulated environment while still allowing for the Jetson TX1 to be used to run the SAE vehicle software in order to maintain as realistic an environment and workload as possible, as the control hardware and software on the simulator is identical to the ones used in the real vehicle. In addition, the simulation node handles input from a Logitech G920 racing wheel [29] and simulates the low-level safety systems in order to allow overriding autonomous functions using manual inputs in a similar manner to what is possible on the SAE vehicle.

VII. SYSTEM VALIDATION

In order to verify our system, we conducted experiments relating to sensor fusion for dead reckoning, waypoint and cone driving, and the driving simulator, which are elaborated individually in Sections VII-A through VII-D.

A. Sensor Fusion

The odometry measurements are compared to the fused odometry and IMU position estimate using our approach described in Section III-B. To gather the data, the vehicle was driven in a relatively straight path on an even plane and as a result the z coordinate was omitted. In Table I, we hence

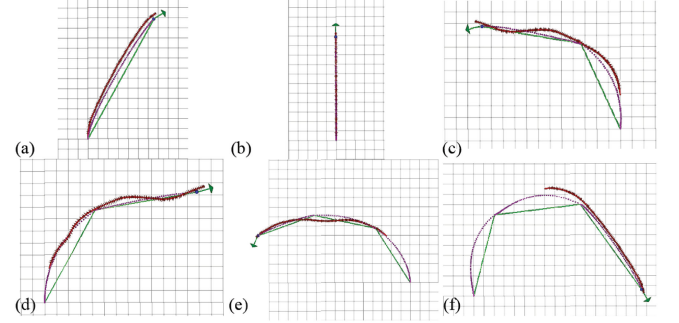
Fig. 9. Generated path projections M_i (red), its ground truth D_i (purple) and the linear displacements of waypoints (green). [Grid size: 1 m \times 1 m]TABLE II
ERROR AND DISTANCE MEASUREMENTS FROM FIG. 9

Fig. 9	ϵ_p (m)	M_i (m)	D_i (m)
(a)	0.316	13.895	13.579
(b)	0.000	10.895	10.895
(c)	0.412	15.053	14.158
(d)	0.368	15.263	14.482
(e)	0.474	13.895	14.105
(f)	0.626	14.053	13.737

measured and calculated the displacement of the car's state Δx_k and its error covariance P_k across three time instances k measured in seconds. We then compared the values obtained through pure wheel odometry (no IMU fusion) against that from the EKF (with IMU fusion); these values are expressed in Cartesian coordinates (x, y) and are measured in metres.

The results in Table I shows an improvement in the certainty of positioning, whereby sensor fusion has performed corrections to the coordinates and improved the covariances in x and y . In our tests, the combination is sufficiently accurate for this application.

B. Waypoint Driving

We carried out our experiments for waypoint driving by measuring its path planning accuracy through the calculation of waypoints across several driving scenarios as shown in Fig. 9. Figure legends are as presented in Section IV-A. This accuracy is quantified by the car's projection error ϵ_p (the maximum deviation of the path projection from the ground truth), and its root-mean-square error ϵ_{rms} .

$$\epsilon_{rms} = \sqrt{\frac{\sum_{i=1}^n (D_i - M_i)^2}{n}} \quad (5)$$

where n denotes the total number of records; D_i and M_i denotes the distance of the ground truth and the projected path, respectively at record i .

Distance error measurements are shown in Table II, while heading angle deviations were found to be insignificant.

Using the values of M_i and D_i in Table II, ϵ_{rms} was calculated to be 0.525 m, which is 85% accurate when compared to the average transverse track width of 3.5 m. This accuracy indicates that D is relatively close to M across the total i records. Additionally, the increase in track complexity (such as through the addition of sharper turns and more segments) contributes to a



Fig. 10. Experimental setup for cone driving.

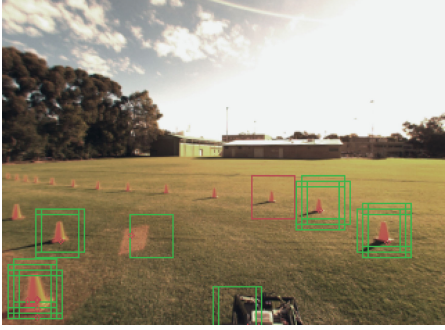


Fig. 11. Visual cone detection showing the detected cones in bounding boxes.

 TABLE III
F₁ SCORES FOR VISUAL CONE DETECTION

Case	Precision	Recall	F ₁
Mean	0.9568	0.7644	0.8499
Best	1.0000	1.0000	1.0000
Worst	0.7500	0.5524	0.6362

greater increase in ϵ_p , as compared to the increase of $(D_i - M_i)$. As expected, ϵ_p is non-existent when a perfectly straight path is generated as shown in Fig. 9(b).

C. Cone Driving

Cone driving on the car follows the setup as prescribed by the standards set by Formula Student Germany's Track Marking and Skidpad for Dynamic Events (DE6.3/DE6.4) [30]. Each cone measures $228 \times 228 \times 325$ ($l \times w \times h$) mm and they are placed as pairs 5 m apart, creating a track width of 3.5 m, as illustrated in Fig. 10.

The system obtains the positions of the cones through the combination of LiDAR and vision processing. Data from the LiDAR is simultaneous to visual cone detection to provide a more robust solution to cone positioning. In our experiments, the speed of the car is limited to 5 m/s due to safety considerations.

To detect cones in the vicinity, we first apply the process described in Section V-C to find a suitable path to navigate. Fig. 11 shows the detected cones in bounding boxes, which are red upon detection turns green as it passes the colour filter. The accuracy of visual cone detection for our initial training and test sets are good, at 96.3% and 92.1% respectively, with its F₁ score as calculated in Table III, which are given in their mean, best

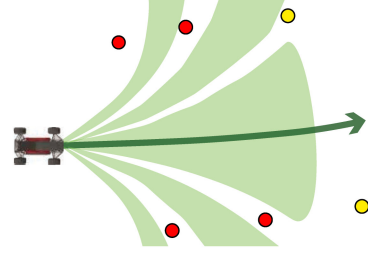


Fig. 12. Visualisation of the path planner on cone driving. Immediate cones are red and subsequent cones are yellow; green regions are viable paths.

 TABLE IV
PATH ERRORS

N	n_e	Σn	e
1	7	84	7.14%
2	3	62	4.84%
3	1	24	4.17%

and worst cases as measured from each frame across different lighting conditions. These results imply that our classifier is highly accurate under certain conditions and with a mean F₁ score of 0.85, our visual cone detection algorithm is therefore deemed suitable for the system.

Meanwhile, measurements from the LiDAR are used to accurately obtain the relative position of the cones. Its accuracy is verified by comparing it against their ground truth distances. We have thus calculated the mean distance error ϵ_d to be 21.30 mm with its standard deviation σ_d at 15.49 mm. By evaluating these results against the 5 m cone distances, we have subsequently deduced that the LiDAR system is adequately accurate for dynamic cone positioning.

Paths are generated through the clustering and filtering of LiDAR data. With reference to Fig. 12, the vehicle will drive straight following the green arrow in the green region as it has the largest range free of objects. Path planning for cone driving was tested across three sets of recordings N to verify the consistency of path generation across all subsequent frames Σn . Frames with false positives and negatives are considered erroneous frames n_e , where the error percentage e is then calculated with our results given as Table IV. With a mean false detection of less than 5% and the erroneous frames at less than 8%. We have therefore deduced from these results that our cone driving algorithm is adequate for autonomous driving. Note that these errors can be further remedied with frame coherence, which is capable of eliminating the classification of stray frames.

The runtime performances of individual nodes were recorded during our experiments, which are given in percentages as their means and standard deviations for CPU (avgCPU, stdCPU) and memory consumption (avgRAM, stdRAM), as tabulated in Table V. The sensors driver nodes make up the largest utilisation percentage, with the image captures occupying over 60% of the CPU footprint to capture a series of 3-channel, 8-bit calibrated RGB image from the camera pair. The LiDARs collectively consume 18% of CPU. The path planning (ConeDetect) and high-level control nodes operate at 20 Hz, with 11% CPU usage.

TABLE V
RUNTIME PERFORMANCES FOR CONE DRIVING

Nodes	avgCPU	stdCPU	avgRAM	stdRAM
Camera	63.47%	3.862%	1.5%	0.0%
IMU	7.157%	0.5182%	1.9%	0.0%
LUX	9.955%	0.6802%	1.7%	0.0%
LMS111	7.586%	0.4830%	0.30%	0.0%
ConeDetect	7.605%	0.7858%	0.38%	0.040%
roscore	0.1354%	0.1909%	0.88%	0.61%
control	3.557%	0.2461%	0.30%	0.0%

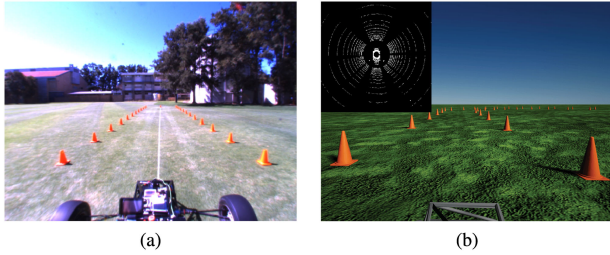


Fig. 13. Scenarios used for comparing (a) real and (b) simulated LiDAR and visual cone processing outputs.

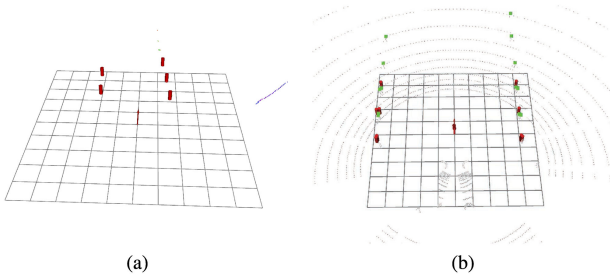


Fig. 14. LiDAR cone processing outputs from (a) real and (b) simulated scenarios.

D. Driving Simulation

The effectiveness of the simulation system was measured through the drawing of comparisons to results gathered from testing of the SAE vehicle. Given the current work involving autonomously driving a traffic cone delineated race track, a focus was placed on the relative accuracy of the LiDAR and visual cone detection systems compared to results gained from test drives of the SAE vehicle. Comparisons were made by recreating a cone track on a flat plane in the simulation system and performing visual comparisons of the results. A comparison of the tracks used is displayed in Fig. 13.

From Fig. 14, the output from the simulated LiDAR is significantly more detailed than that on the SAE vehicle. As such, the original LiDAR output from the simulator (white points) was cropped to simulate the LiDAR available on the SAE vehicle through the use of ROS' "pointcloud_to_laserscan" package, resulting in the 2D laser scan data displayed in green. While these laser scans are not identical, with the real data displaying the ground on the far right as a result, of the uneven terrain, these figures show that the cone locations identified are sufficiently similar to allow testing of higher level components (e.g. path planning) on the simulated system.

VIII. CONCLUSION

We have presented a software framework for a high-level control system that is designed for autonomous vehicles that is both modular and scalable. Our design approach using open-source software with commercially available sensors and parts hopes to encourage similar projects especially in academia where we have fitted a student competition vehicle for full autonomous driving. These projects can therefore be low-cost while allowing users to adapt the software to the vehicle's and environment's needs. This system aims to be holistic by incorporating all the necessary modules required for autonomous driving, including sensor interfaces and fusion, localisation, path planning, visual navigation and road detection; as well as cone driving and an identical driving simulator for both real-world and simulated tests. It is therefore easily deployable while requiring minimal configuration. Experiments on path planning, cone driving and the simulator proved that this system is robust and adequate for implementation. We are eager to correspond with any entities who wish to incorporate our approach in their respective projects.

ACKNOWLEDGMENT

The authors would like to thank Nvidia Corporation for providing the GPU and the Jetson TX1 under its GPU Grant Program; along with REV Sponsors — Synergy, Galaxy, Xsens, and Altronics.

REFERENCES

- [1] SAE International, "Student Events - Events - Collegiate Design Series." [Online]. Available: <https://www.sae.org/attend/student-events/>. Accessed on: May 29, 2019.
- [2] T. H. Drage, "Development of a Navigation Control System for an Autonomous Formula SAE-Electric Race Car," Honours Thesis, The University of Western Australia, 2013. [Online]. Available: <http://robotics.ee.uwa.edu.au/theses/2013-REV-Navigation-Drage.pdf>
- [3] K. L. Lim *et al.*, "A modular software framework for autonomous vehicles," *2018 IEEE Intell. Vehicles Symp.*, Changshu, China, pp. 1780–1785, Jun. 2018, doi: [10.1109/IVS.2018.8500474](https://doi.org/10.1109/IVS.2018.8500474).
- [4] Open Source Robotics Foundation, "ROS/Introduction - ROS Wiki." [Online]. Available: <http://wiki.ros.org/ROS/Introduction>. Accessed on: May 29, 2019.
- [5] Baidu, "Apollo." [Online]. Available: <http://apollo.auto/>. Accessed on: May 29, 2019.
- [6] Autware, "Autware." [Online]. Available: <https://autware.ai/>. Accessed on: May 29, 2019.
- [7] Google Developers, "Protocol Buffers." [Online]. Available: <https://developers.google.com/protocol-buffers/>. Accessed on: May 29, 2019.
- [8] Arduino, "Arduino Nano." [Online]. Available: <https://store.arduino.cc/arduino-nano>. Accessed on: May 29, 2019.
- [9] Xsens, "MTi-G-710." [Online]. Available: <https://www.xsens.com/products/mti-g-710/>. Accessed on: May 29, 2019.
- [10] B. M. Yu, K. V. Shenoy, and M. Sahani, "Derivation of Extended Kalman Filtering and Smoothing Equations," Oct. 2004 [Online]. Available: http://www-npl.stanford.edu/byronyu/papers/derive_eks.pdf. Accessed: May 9, 2018.
- [11] D. Morrell, "Extended Kalman Filter Lecture Notes," EEE 581-Spring, Arizona State University, Tempe, Arizona, 1997.
- [12] T. Moore and D. Stouch, "A generalized extended kalman filter implementation for the robot operating system," in *Proc. Intell. Auton. Syst.* (13, ser. Advances in Intelligent Systems and Computing), E. Menegatti, N. Michael, K. Berns, and H. Yamaguchi, Eds. New York, NY, USA: Springer, 2016, pp. 335–348.
- [13] SICK AG, "LMS111-10100." [Online]. Available: <https://www.sick.com/au/en/detection-and-ranging-solutions/2d-lidar-sensors/lms1xx/lms111-10100/p109842>. Accessed on: May 29, 2019.

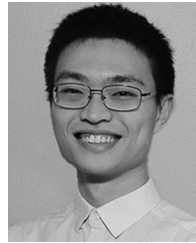
- [14] AutonomouStuff, “ibeo Standard Four Layer Multi-Echo LUX Sensor | LiDAR | Product,” [Online]. Available: <https://autonomoustuff.com/product/ibeo-lux-standard/>. Accessed on: May 29, 2019.
- [15] T. Drage, T. Churack, and T. Braunl, “LIDAR road edge detection by heuristic evaluation of many linear regressions,” in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, Sep. 2015, pp. 2465–2470.
- [16] FLIR, “Blackfly 1.3 MP Color GigE PoE (Sony ICX445),” [Online]. Available: <https://www.ptgrey.com/blackfly-13-mp-color-gige-vision-poe-sony-icx445-camera>. Accessed on: May 29, 2019.
- [17] FLIR, “Fujinon YV2.82.8sa-2, 2.8mm-8mm, 1/3,” CS mount Lens,” [Online]. Available: <https://www.ptgrey.com/fujinon-yv28x28sa-2-hd-vari-focal-lens-3>. Accessed on: May 29, 2019.
- [18] C. Liang, L. Chang, and H. H. Chen, “Analysis and compensation of rolling shutter effect,” *IEEE Trans. Image Process.*, vol. 17, no. 8, pp. 1323–1330, Aug. 2008.
- [19] H. R. Kam, S.-H. Lee, T. Park, and C.-H. Kim, “RViz: A toolkit for real domain data visualization,” *Telecommun. Syst.*, vol. 60, no. 2, pp. 337–345, Oct. 2015. [Online]. Available: <http://dx.doi.org/10.1007/s11235-015-0034-5>
- [20] C. De Boor, *A Practical Guide to Splines*. New York, NY, USA: Springer-Verlag, 1978, vol. 27.
- [21] OpenCV, “Opencv,” 2018. [Online]. Available: <http://opencv.org/>
- [22] V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: A deep convolutional encoder-decoder architecture for image segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.
- [23] K. L. Lim, T. Drage, and T. Braunl, “Implementation of semantic segmentation for road and lane detection on an autonomous ground vehicle with LIDAR,” in *Proc. IEEE Int. Conf. Multisensor Fusion Integration Intell. Syst.*, Daegu, Nov. 2017, pp. 429–434.
- [24] R. Mur-Artal and J. D. Tardós, “ORB-SLAM2: An Open-Source SLAM system for monocular, stereo, and RGB-D cameras,” *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.
- [25] NVIDIA Corporation, “Parallel programming and computing platform—cuda,” 2018. [Online]. Available: <http://www.nvidia.com/cuda>
- [26] Y. Chen, “yunchih/ORB-SLAM2-GPU2016-final,” Feb. 2019. [Online]. Available: <https://github.com/yunchih/ORB-SLAM2-GPU2016-final>
- [27] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Region-Based convolutional networks for accurate object detection and segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 142–158, Jan. 2016.
- [28] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Proc. 1st Annu. Conf. Robot Learn.*, 2017, pp. 1–16.
- [29] “G920 driving force racing wheel for box one and pc,” [Online]. Available: <https://www.logitechg.com/en-au/product/g920-driving-force>. Accessed on: May 29, 2019.
- [30] Formula Student Germany GmbH, *FSG Competition Handbook 2018*, Wiesbaden, Germany, Apr. 2018. [Online]. Available: <https://www.formlastudent.de/fsg/rules/>. Accessed on: May 29, 2019.



Kai Li Lim received the B.Eng. (Hons) degree in electronic and computer engineering from the University of Nottingham, Nottingham, U.K. in 2012, and the M.Sc. degree in computer science from Lancaster University, Lancaster, U.K. in 2014. He is currently working toward the Ph.D. degree at The University of Western Australia, Perth, Australia, sponsored by the Australian Government under the Research Training Program. His research interests include visual navigation, navigational algorithms, and electric vehicles.



Thomas Drage received the B.Eng. (Hons) degree with in electrical and electronic engineering and the B.Sc. in physics in 2014 from The University of Western Australia, Perth, Australia, where he is currently working toward the Ph.D. degree. He has worked in the fields of industrial instrumentation and process control for the Oil and Gas industry. His research interests include autonomous vehicles, in particular, functional safety, data science applications, and advanced sensor systems.



Chao Zhang received the B.Sc. degree in earth science information and technology from the Sun Yat-Sen University, Guangzhou, China, in 2014. He is currently working toward the M.P.E. degree in electrical and electronic Engineering at The University of Western Australia, Perth, Australia. His research interests include computer vision, machine learning, and autonomous driving.



Craig Brogle received the B.Sc. degree in engineering science and physics in 2015 from The University of Western Australia, Perth, Australia, where he is currently working toward the M.P.E. degree in software engineering. His research interests include software architecture and autonomous driving.



William W. L. Lai received the B.Sc. degree in computer science from Edith Cowan University, Perth, Australia, in 2003. He is currently working toward the M.P.E. degree in electrical and electronic engineering from The University of Western Australia, Perth, Australia. His research interests include electric vehicles and autonomous driving.



Timothy Kelliher received the B.Sc. degree in engineering science and economics in 2017 from The University of Western Australia, Perth, Australia, where he is currently working toward the M.P.E. degree in electrical and electronic engineering. His research interests include electric vehicles and autonomous driving.



Manuchekhr Adina-Zada received the B.Sc. degree in engineering science in 2016 from The University of Western Australia, Perth, Australia, where he is currently working toward the M.P.E. degree in electrical and electronic engineering. His research interests include electric vehicles and autonomous driving.



Thomas Bräunl received the Diploma from the University of Kaiserslautern, Kaiserslautern, Germany, the M.S. degree from the University of Southern California, Los Angeles, CA, USA, and the Ph.D. and Habilitation degrees from the University of Stuttgart Stuttgart, Germany. He was the Technical Director of the West Australian Electric Vehicle Trial and worked on driver-assistance systems with Daimler/Mercedes-Benz, Stuttgart, and on electric vehicle charging systems with BMW, Munich and Mountain View, CA. He is a Professor with UWA, Perth, Australia.

CHAPTER 6

Integrated Modular Safety System Design for Intelligent Autonomous Vehicles

Integrated Modular Safety System Design for Intelligent Autonomous Vehicles

Thomas Drage, Kai Li Lim, Joey En Hai Koh, David Gregory, Craig Brogle and Thomas Bräunl

Abstract—This paper presents an approach to specifying a modularised safety system which comprehensively addresses the safety requirements for highly autonomous (SAE Level 3+) road vehicles featuring advanced sensing and automated navigation. As these requirements are often overlooked in similar autonomous driving system proposals, we present a method of hazard and risk analysis which investigates hardware failures, environmental perception limitations, human interaction and functional requirements for artificial intelligence. We then define a system design which implements the required safeguards and examines the application on two electric autonomous vehicle testbeds: a race car and a shuttle bus. The close-coupling of a safety-oriented architecture and multi-regime Hazard and Risk Assessment process was tested to measure the system's ability to detect and react to pedestrian stimuli, resulting in accurate detections and reactions, thereby confirming its ability to design safety systems for autonomous research vehicles in a scalable and easily assured fashion.

I. INTRODUCTION

The Renewable Energy Vehicle Project (REV) at the University of Western Australia conducts research into electric vehicles, vehicle automation and autonomous driving systems. Recent projects, shown in Fig. I, include the development of an Autonomous Formula-SAE Electric car [1], an open-wheeled, electric drive race car, with electronic drive-by-wire and electromechanical brake/steering actuation. The vehicle serves as a compact, flexible test-bed for sensor testing and the development of autonomous driving algorithms. The group's current focus is the high-level automation of a passenger shuttle bus, using an electric drive-by-wire platform from bus manufacturer Ligier, but with our own navigation system. The shuttle will operate as a self-driving people mover on campus and will be flexible enough to dynamically plan its route. All sensory and navigation processing is on-board; there will be no dependence on cellular networks or other high-bandwidth communication systems or remote servers.

Development of Level 3+ [2] Autonomous Driving Systems (ADS) presents a significant risk to both people and infrastructure due to the requirement for complex, software driven electromechanical systems to now provide safe driving behaviour under normal conditions in complex and changing environments [3]. Indeed, whilst Autonomous Vehicles have been heralded with promises of improved traffic safety and lower collision rates, current technology may not offer these

advantages and significant progress is required in the realms of safety and reliability, with disengagement of autonomous systems, requiring resumption of manual control to achieve safety still relatively common [4]. With technologies improving rapidly, the situation is expected to improve. However, like the critical systems used for aeronautical control, there is significant room for improvement of the processes used to assure safe performance.



Fig. 1. REV vehicles: F-SAE and nUWay shuttle.

Historically, the only established industry standard for safety of the electronic/software systems' underlying vehicle automation was ISO 26262, "Road vehicles - Functional safety", itself a derivative of IEC 61508, the parent standard concerning "Functional Safety of Electrical/Electronic/Programmable (E/E/P) Electronic Safety-related Systems" [5]. Other derivatives are specifically targeted at Machinery Safety (IEC 62061), Process Industries (IEC 61511), etc. Such standards focus on the application of a safety life cycle for control of systematic failures and the probabilistic assessment of random hardware failures [6].

Recently, this suite has been joined by ISO/PAS 21448:2019, the first release of the standard "Road Vehicles — Safety of the intended functionality". This guides design and verification/validation, targeted at Level 1 and 2 Advanced Driver Assistance System (ADAS) and ADS, but with scope for application to higher levels. The concept does not focus on hardware failure, but instead on "functional insufficiencies of the intended functionality" [7], which include issues relating to environmental perception and reasonably foreseeable misuse of ADS. However, further research must consider approaches beyond these standards and evaluate the safety of driving behaviours themselves [3] and the multitude of cross-disciplinary aspects relating to safety, from electronic/software design to a legal framework required to assure the safety of an autonomous or partially autonomous road transportation system [8].

Common to both the regimes of Functional Safety (FS) and Safety of the Intended Functionality (SOTIF) are the

The authors are with The REV Project, School of Electrical, Electronic and Computer Engineering, The University of Western Australia, Perth WA 6009, Australia. thomas.drage@research.uwa.edu.au, {kaili.lim, thomas.braunl}@uwa.edu.au

accurate identification and risk assessment of hazards. By extension, this process is also applicable to what we may define here, as the Safety of the Artificial Intelligence (SOTAI), which encompasses dynamically planned driving behaviours in variable environments and other such real-time decision making which are carried out when there are no faults and no unintended loss of functionality. Both standards give allowance for application of a variety of techniques for Hazard and Risk Assessment (HARA) and various proposals have been made for combination of such activities [9], [10]. Such techniques include Preliminary Hazard Analysis (PHA), Hazard and Operability Studies (HAZOP), Failure Mode and Effects Analysis (FMEA), Fault Tree Analysis (FTA) and have been applied to various aspects of robotics [11], [12] as well as to FS [13]. More recently, extension of these techniques, by increasing generality [13], [14] and with new techniques such as System Theoretical Process Analysis (STPA) [15] aimed at improved capture of the types of hazards associated with SOTIF/SOTAI in addition to FS (that is those outside of the realm of E/E/P system faults).

In this paper, we propose an iterative application similar to [14] of a generalised, HARA based upon the HAZOP methodology, but with a quantitative approach [16]. This is intended to align with the Automotive Safety Integrity Level (ASIL) concept of ISO 26262, which defines the required component reliability for FS. Where the ASIL concept itself does not extend well to the type of safeguards associated with SOTIF/SOTAI, we propose the Autonomous Vehicle Layer of Protection Analysis (AV-LOPA) concept. This method is based upon the LOPA methodology which has recently emerged as accepted best practice in the process industry [17] and is exemplified for application to FS in IEC 61508, Part 5 [6]. We adapt and apply this approach to unify the safety regimes in a fashion which gives appropriate credit to high-level safeguards, such as Machine Learning (ML) based algorithms which are not well dealt with within FS [18]. At the time of writing, our proposed system is novel whereby we have identified no other proposals that clearly integrates the safety concerns with a reference architecture; our proposal addresses this across a full range of issues. At the time of writing, our system is novel whereby we have identified no other proposals that clearly integrates the safety concerns with a reference architecture; our proposal addresses this across a full range of issues.

The safeguards required to manage risk in a Level 3+ ADS may consist of systems or modules which include fault-detection interlocks governed by FS, supervisory monitoring systems to achieve SOTIF and advanced or redundant algorithms to achieve SOTAI [19], [20]. Our vehicles feature Robot Operating System (ROS) centred controllers which form part of the control and safeguarding system [21], a method which has been shown to be able to be made compliant with FS [22]. We additionally define a complete system architecture with control and safeguarding hardware and software independence and fail-safe design principles which draw from the typical industrial implementation of

IEC 61508/61511 compliant systems.

It should be noted however, that at present we have considered specialised automotive systems; a race car utilised for research purposes and a shuttle bus which is constrained to drive in a pedestrianised area. In these situations we consider no contribution to risk reduction attributed to a driver and take the approach of identifying risk of failure of components and systems to function as intended. The safe-action in these scenarios is typically non-complex (e.g. emergency stop) and further work is required to apply this approach to systems with complex requirements for safe-actions, as may be found in road traffic systems.

II. HAZARD AND RISK ANALYSIS

For risk assessment of our autonomous driving projects, we utilise a three step process shown in Fig. 2, consisting of vehicle and ADS systemisation, then HAZOP-style hazard identification and followed by a semi-quantitative risk assessment (AV-LOPA) which is used to drive specification of safeguarding requirements. Due to the experimental nature of our vehicle systems and iterative design approach, this process is repeated as features and modifications take place, including the additions of safeguards themselves. All of these activities are performed as a cross-functional team, in a workshop environment in order to provide breadth of experience and knowledge, particularly during hazard identification. By systematic application, failure to identify hazards created by new functionality is minimised.

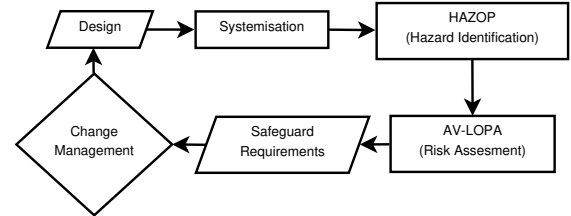


Fig. 2. HARA Process Flow.

Identification of hazards associated with the Autonomous F-SAE car was based on the scenario of speed-limited autonomous driving in an open area with members of the public nearby. Additionally, the HARA was performed considering the scenario with and without a safety driver seated in the vehicle and the safety requirements relating to protecting the driver were assessed in addition to the change in risk of uncontrolled collisions with the external environment. The shuttle's HARA was performed to identify gaps in existing systems and has been iterated as level of automation has increased.

A. Systemisation

The first step involves identification of the systems and subsystems which the autonomous vehicle comprises to guide the scope of the hazard identification. In order to address the requirements of the three safety regimes previously defined, the systemisation must identify both physical and logical components of the vehicle and be extended in such a

way that interconnections between systems are also captured. This step may be considered to take the same place as *Item Definition* within ISO 26262.

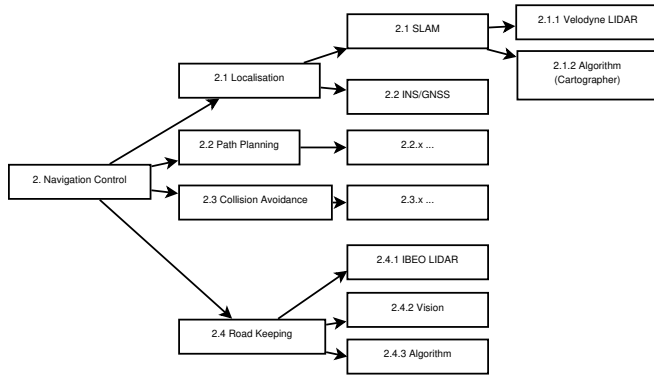


Fig. 3. Extract from High-Level Systemisation.

The intent of this systemisation is to ensure that each of the lowest-level subsystems responsible for providing functionality is examined in order to consider failure to provide function to the systems that depend on them. The intent is not to decompose components, thus systemisation is complete at the assembly or functional level. Detailed examination of the effects of component failure may be undertaken in parallel, via a bottom up process such as FMEA. An example is given for the Navigation Control System of the shuttle bus in Fig. 3 and identifies key functions and dependencies of the system which is a required input to HARA. Consideration should be given to clearly identifying components which multiple systems rely upon.

B. HAZOP

The HAZOP process involves a guided “brainstorming” workshop, which examines each system’s parameters in the context of predefined guide words. The guide words prompt the creation of deviations which could lead to some unsafe scenario which requires risk assessment. We define a base set of parameters and guide-words for this domain:

- **Parameters:** Speed, Error, Surface, Prediction, Rain, Control action, Fog/Dust/Visibility, Inaccuracy, Snow, Driver, Temperature/Heat, Passenger, Sun/Light, Map/Path, Traffic, Surroundings, Pedestrian, Communication/Signal, Obstruction, Pressure, Braking, Voltage, Turning, Maintenance, Banking, Stopping, Starting, Signals, Failure, Fault
- **Guide words:** More, Less, None, Reverse, Unexpected, Undetected, Abnormal, Repeated, Altered, Early, Late, Other than

It is through the definition of these parameters that we can ensure the analysis considers the high-level hazards associated with the SOTIF and SOTAI regimes. The definition of the parameters and guide words is application specific; whilst ours are tailored to our specialised automotive systems, other applications e.g. racing or road passenger traffic would need adjustment of the composition of the set to ensure the correct

balance of coverage and generality. It should be noted that in order to assess all combinations, a systematic method is required to ensure that the process proceeds efficiently but without failure to identify a potential hazard.

Considering, for example, System 2.4.1 per Fig. 3, applying the guidewords to the parameters we can document several of the possible deviations:

- 1) **Less voltage** causing shutdown of lidar and inability to sense/keep to road.
- 2) **More speed** leading to loss of quality perception due to limited scan rate of the lidar.
- 3) **More rain / less visibility** leading to interference with lidar sensing and inability to determine road edge.
- 4) **Abnormal traffic** leading to road keeping no longer being the most safe regime (e.g. another car in oncoming lane).

Of these identified hazards, 1 can be categorised as lying primarily within the FS regime, 2–3 in the SOTIF regime and 4 in the SOTAI regime; thus it is possible to examine multiple aspects through this integrated methodology.

III. LAYER OF PROTECTION ANALYSIS CONCEPT

Each autonomous vehicle in the REV group has several layers of safety. At the lowest level is always a non-software and non-computer based safety system that exclusively utilizes simple and reliable automotive electronics components, such as relays. An example for this is an Electric Vehicle (EV) safety system; which at the lowest level may feature a dead man’s switch, requiring the driver to exhibit certain sensed behaviours, in order for the car to continue driving — or in the absence of a driver, the regular reception of a wireless heartbeat signal. Layered on top of this is an electronically controlled safety system which monitors physical parameters and a supervisory system which ensures functionality of the high-level automation itself. By performing risk analysis we can sum risk-reductions by such safeguards for each hazard scenario identified.

Motivating this approach is the difficulty in applying the ASIL concept of ISO 26262 to highly complex, L3+ systems which are designed by default to have no driver. Assignment of ASIL levels is dependent on the consequence, probability of exposure and additionally the controllability, which refers to the ability for human intervention to rectify a situation in view of system failure. This results in two issues: naive application of the ASIL classification assuming all scenarios are uncontrollable results in the requirement for high ASIL of high-level control systems and secondly it fails to credit redundant systems which can provide Independent Protection Layers (IPL). Furthermore, the ASIL concept itself does not sufficiently address the SOTIF or SOTAI regimes and hence a method is required to provide assessment of safeguards required to ensure these.

Each independent protection layer must fulfil the following criteria:

- 1) **Independence:** It should not have failure modes (e.g. shared sensors) common to other safeguards used in the scenario.

- 2) **Effectiveness:** Each IPL must be able to fully mitigate the hazard.
- 3) **Validation:** Each IPLs functionality must be able to be assured and maintained.

Assurance of independence is key to safety system design, but often difficult to achieve. An example may be considered whereby a vehicle encounters rain which affects control by lidar road sensing and an action is taken to stop the vehicle, however, the braking system is equally degraded by the presence of rain and thus ineffective. In order to achieve SOTIF in this instance, we consider another safeguard which is effective and independent to the lidar and braking system; e.g. automatic throttle limitation according to conditions to ensure control in adverse conditions.

The risk assessment process proceeds then in three steps - firstly an unmitigated risk score is qualitatively calculated ($\text{InitialRisk} = \text{Likelihood} \times \text{Consequence} \times \text{Exposure}$) using a risk-matrix of the type suggested by IEC 61508 (for HARA) [6] and MIL-STD-882E (for PHA) [23], shown in Table I. Thus, a **likely** event such as a distracted pedestrian on campus, which gives **infrequent** exposure to the consequence (e.g. pedestrian is struck) with very serious consequence (e.g. death) gives a risk rank of **very high**.

TABLE I
RISK MATRIX

Likelihood (<i>L</i>)		Exposure (<i>E</i>)	
Almost certain	10,000	Continuous	10
Likely	1,000	Frequent	5
Unusual	100	Occasional	3
Remote possibility	10	Infrequent	2
Conceivable	1	Rare	1
Practically impossible	0.1	Unheard of	0.5

Consequence	Risk Rank					
Catastrophe	M	H	VH	VH	VH	VH
Disaster	L	M	H	VH	VH	VH
Very serious	L	M	H	VH	VH	VH
Serious	L	L	M	H	VH	VH
Important	L	L	L	M	H	VH
Noticeable	L	L	L	L	L	M
$L \times E$	<1	<10	<100	<1k	<10k	<100k
	VERY HIGH	HIGH	MEDIUM	LOW		

Note that the calibration of such tables is dependent upon organisational or regulatory risk tolerance and is not discussed here, except to note that recommendations are given in various local and international standards (e.g. MIL-STD-882E). Next, using this risk score, the amount of risk reduction can be calculated by determining the factor of reduction in the score required to achieve acceptable risk, which in this case is defined as low or if not reasonably practicable, medium.

Next we list the layers of protection available and combine them. Formally, this would be multiplication of the safeguards probability of failure per unit time, i.e.

$$f_{\text{event}} = f_{\text{unmitigated}} \times P_1 \times P_2 \times \dots \times P_n \quad (1)$$

where f is the event frequency and the P_n are the probability of failure per unit time of each safeguard. However, here we apply a semi-quantitative approach and define a system whereby order-of-magnitude estimates are used to define the integrity of a layer as a credit value allowing a simple (logarithmic) sum to determine the overall integrity of the layers of protection. To this end, we can assign such credits as shown in Table II.

TABLE II
EXAMPLE IPL CREDIT GUIDELINES

Description	Order of Likelihood Reduction	Credit Value
ASIL A System	10	1
ASIL B/C System	100	2
ASIL D System	1,000	3
Human safety driver present but not ensured to be engaged	10	1
Human safety driver with dead man's switch	100	2
Physical speed limiter	10	1
Supervisory software	10	1
Geofencing system	10	1
Procedural control	10	1

Assignment of these credit values can be via quantitative methods (e.g. FMEA), through analysis of component failure rates (e.g. proven-in-use concept of IEC 61508) or can be experience-based for lower-level integrity safeguards; the listing in Table III is not exhaustive and should be considered based on available technologies. Note that our mapping for ASIL likelihood reduction is conservative, encouraging the use of multiple IPLs to achieve the required integrity of safeguarding. Note that adding IPLs relating to human safety drivers requires that ASILs are evaluated with C3 controllability class.

Considering further the example of a hazard caused by a distracted pedestrian in the vehicle's path, we calculate the risk level to be *very high* and must therefore consider at least three orders of magnitude risk reduction. We construct IPLs as shown in Table III. Therefore, in this instance we consider the residual risk to be low and acceptable mitigations present to allow operation. The hazard identification and AV-LOPA assessment is recorded in a tabular format and made available for reference when planning autonomous vehicle operations.

TABLE III
AV-LOPA EXAMPLE

IPL	Credit
Free space optimisation algorithm avoids paths through regions with high pedestrian risk	1 (preventative)
Computer vision distracted pedestrian detector system identifies hazard	1 (mitigative)
Path planning system detects pedestrian obstacle	1 (mitigative)
Safety curtain lidar stops bus via emergency brake if object obstructs bus	1 (mitigative)
TOTAL	4

IV. SYSTEM DESIGN

A. Autonomous F-SAE Race Car

The safety system implemented for the Autonomous F-SAE Race Car [24] is partitioned into functionality located across three logic solvers; the navigation controller (a ROS-based computer system), the hardware safety supervisor and the drive-by-wire controller. The latter two are both considered components of the low-level system, directly interfacing with sensors/actuators and are based upon a combination of rule based and state-machine logic. Fig. 4 details the system architecture, which uses high and low-level supervisory systems to facilitate implementation of safeguards across the three safety regimes defined in Section I.

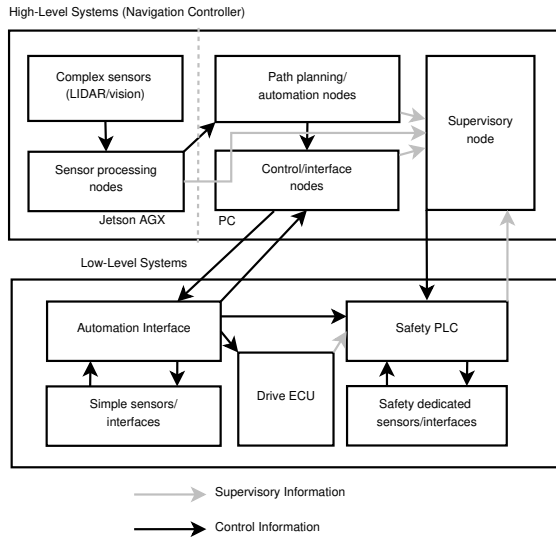


Fig. 4. System design overview.

The safety supervisor is critical to our system design and firstly implements detection of attempts by a safety driver to override the drive-by-wire system via the steering wheel and brake pedal. Secondly it implements sequencing to ensure that the autonomous system is only able to be activated when the emergency shutdown system is enabled and provides interlocks against attempts to activate it while the vehicle is not in a safe state. It also provides an interface for the navigation controller to provide safety functionality and monitoring of the drive-by-wire and navigation controllers.

Thus, we can consider that all three aspects of safety engineering are associated with this component:

- FS is applicable as it is an E/E/P device responsible for safeguarding against hazardous events, e.g. disconnection of motor power in case of driver override of brake pedal.
- By virtue of its supervisory functionality and safeguarding against misuse it implements SOTIF.
- By providing an interface for the navigation controller's safety module and the ability to consider redundant or alternative sensor inputs, it provides the ability to implement SOTAI safeguards that take mitigative action.

B. nUWay Shuttle Bus

The nUWay shuttle bus system architecture follows that of the prior F-SAE vehicle project, modified to suit the architecture of the commercial vehicle. In addition to the ECUs used in the drive control system, the shuttle bus features an industrial safety PLC and computer based high-level navigation system, connected to a single CAN network. The safety PLC provided by the shuttle manufacturer handles basic sequence-control of the vehicle (e.g., interlocking doors and brakes) and provides action for fault conditions and detections from the perimeter lidars, which implement a simple distance-based safety curtain.

The high-level automation systems in our shuttle bus comprise two main computers; an industrial x86 PC (manufacturer provided) for handling IO and core navigational functionality, and an NVIDIA Jetson AGX Xavier (installed by our project) which provides accelerated execution of Deep Learning based algorithms used for interpreting sensor data. ROS is used once again to host the node-oriented high-level system.

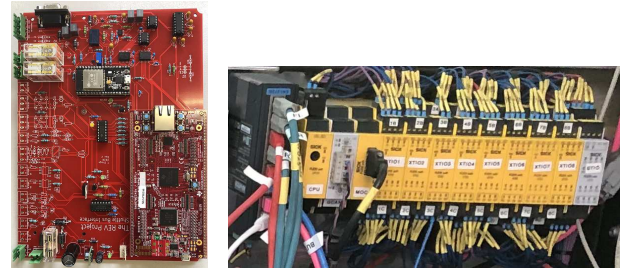


Fig. 5. nUWay system interface controller and low-level PLC control and safety system.

As we were not able to obtain the vehicle's CAN bus specification for driving and steering, we developed an interface system based around a TI Hercules automotive ARM microcontroller, which is certified for ASIL applications (Fig. 5). This also allows us to implement additional drive-control and safety features as part of our automation system outside of the "black box" shuttle bus ECU. Additionally, it provides means for use of a wireless Bluetooth hand-controller for manual manoeuvring of the vehicle and adds a hard-wired interface to the safety systems, providing redundancy. Critical interfaces are implemented using isolated buses with loopback circuits for constant error checking, ensuring that any abnormal condition will stop the vehicle.

V. MULTI-LEVEL SAFETY SOLUTION

The nUWay shuttle bus implements several layers of safety systems, designed to ensure that fully-autonomous operation in an environment shared with pedestrians is safe and efficient at all times. Our project has extended the safety functionality beyond low-level hazard mitigation systems and addresses the SOTIF and SOTAI regimes, as shown in Fig. 6.

A. Low-Level Safety Features in Autonomous Shuttle Bus

The electric shuttle bus features some built-in low-level safety features, however, in-depth HARA was required for

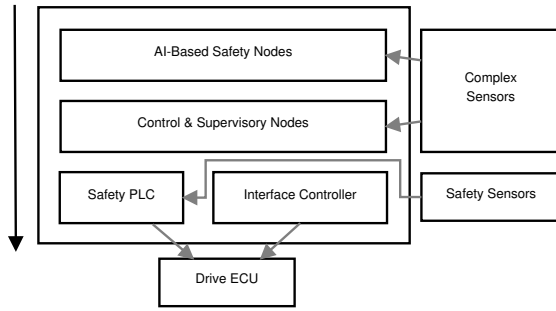


Fig. 6. nUWay shuttle multi-level safety architecture.

our application, which is currently under testing for public usage as a fully autonomous, and eventually unsupervised vehicle. Using the process proposed in this paper, we evaluated the suitability of the shuttle and our navigation system. The hardware safety features present already in the vehicle already include [25]:

- 1) **Obstacle detection:** Four single-beam lidar sensors form a detection area at a 30 cm height above ground level and secures a 2 m collision-free safety zone around the vehicle. However, this leaves the vehicle blind to any obstacles lower than 30 cm.
- 2) **Emergency Stops (E-stops):** Pushing one of the four in-vehicle stop buttons will immediately stop the vehicle and disarm the doors.
- 3) **SOS button (intercom):** This in-cabin button is typically used to establish a communication link to a remote operator.
- 4) **Redundant braking systems:** The vehicle features multiple braking methods, including a fail-safe emergency brake.

These functions are implemented by use of an industrial safety PLC shown in Fig. 5, which operates independently of the navigation system and interconnects with the drive-control ECUs for speed detection etc. The PLC also facilitates auxiliary sequencing functions including lighting and door control. This approach provides robustness in this application and is well suited to the specialised vehicle, however, whilst an architecture utilising an independent high reliability system generalises to other types of vehicles, other applications may require hardware more suited to safeguarding via continuous or higher-level control which are examined in ISO 26262.

The manufacturer provides a joystick for manually operating the shuttle. However, with the shuttle having a symmetric design, there is significant risk of unintended driving in the incorrect/opposite direction unless the joystick unit is correctly rotated. To address this hazard a wireless hand-controller with fault detection system and dead-man's switch was implemented.

B. High-Level Safety Requirements for Autonomous Shuttle Bus

Commercial autonomous shuttles are typically designed for continuous supervision, either via an alert safety officer

on-board the shuttle at all times or a constantly present remote supervisor. They are responsible to stop the vehicle to avoid a possible collision, manually drive the vehicle around an obstacle, re-arm the vehicle and re-initialise autonomous mode after the vehicle has stopped in front of an obstacle or if an e-stop button has been pressed. Our project intends to develop an intelligent autonomous vehicle with the capability for dynamic path planning between stops which would reduce the reliance upon human intervention. This results in increased risk level and our goal of eventually removing the intensive human supervision as a safeguard, implies that a greater number of high-integrity high-level IPLs will be required.

Therefore as we continue to increase the level of automation of the shuttle, we iteratively apply the HARA process. Initially, SOTIF concerns were addressed, particularly relating to the coverage of the safety lidar system and the manual operator control interface. As full automation is achieved, using the extensive sensor suite available (eight lidars, four cameras, INS/GNSS), we implemented a hardware safety supervisor system (via our interface controller) and high-level supervision software (via ROS node), similar to that of the F-SAE car, which aid in our development of safeguards for SOTIF and SOTAI related hazards. Below, we present two high level safety functions which form our multi-level system, developed in response to our HARA.

C. Embedding Morality and Temporal Context in Free Space Detection

Free Space Detection is an important component in Autonomous Vehicle (AV) systems. It involves continuous detection of drivable unoccupied space surrounding the AV. Most research focuses on faster or more accurate estimation of free space — with a research gap in the contextual differences of free spaces. There are two additional context layers that we sought to embed into the free space detection problem. In Fig. 7:

- 1) **Temporal:** Predicted motion trajectories (red arrows) probabilities of traffic agents. Reflected through training a neural network based on Mathieu, Coupire and LeCun's method of future video frame prediction [26].
- 2) **Moral:** Importance attached to traffic agent's life. In this case, human life is valued over dog life as an example. Reflected through semantic segmentation of classes and attaching a numerical 'consideration' weight to each.

Trolley problem scenarios are hypothetical thought experiments as an AV's intelligent systems should prevent the situation of being forced to choose between two fatal outcomes. However, any route planned by the AV changes the risk allocation towards all nearby traffic agents i.e. driving closer to someone poses more risk to them than away. Thus, the viability of free space is a concept explored here. Firstly, an area of current free space that will be occupied in the next time instance is less viable than current free space that will remain unoccupied. Secondly, consider an elderly person, from assumed lower reaction speed and mobility —

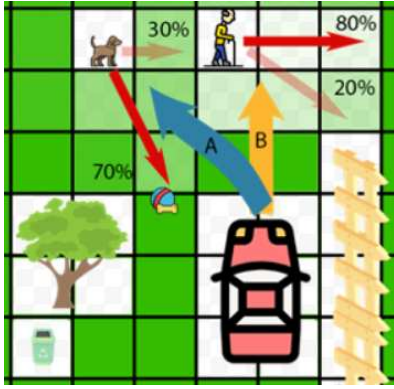


Fig. 7. AV deciding between Path A and B, risking collision between dog and elderly respectively. Free space viability represented by green intensity and influenced by context.

human drivers would give more consideration and slow down more than if it was a young adult crossing in front. Hence, we reflect this difference in moral consideration through the viability of free space around a traffic agent based on their class.

Our methodology involves applying semantic segmentation to 360 degree camera data from four colour cameras (front, back, left, right). Thereafter we use inverse perspective mapping to combine and transform those different viewpoints into semantic segmented Bird's Eye View (BEV) images. Inverse perspective mapping does not work well with irregularities found in real world environments and objects. Thus, the semantic segmentation step is crucial in smoothing out many irregularities. This approach is based on Reiher's approach, and a neural network further corrects the BEV images [27].

With traffic agent classes embedded through semantic segmented BEV images, and future frame prediction of that — an equation with weights assigned to each class accounts for both aspects, and outputs viability scores of each unit of free space. This augmented version of free space acts as an assistive safety layer within our SOTAI regime by constraining the path planning AI to optimise safety.

D. Pedestrian Collision Safety System

Avoiding distracted pedestrians can present a fall hazard to passengers on-board the shuttle bus due to the onset of sudden braking and lack of passenger restraints when low level safety systems mitigate the risk of a collision with other objects and road users by emergency stopping the vehicle. They may pose a risk to the bus if they alter their path suddenly or do not actively try to avoid a collision with the bus. On a university campus, mobile phone distracted pedestrians pose a high collision risk against vehicles which is evidenced in a large increase in injuries in such scenarios [28], [29]. The high level pedestrian collision safety system is implemented with a machine learning approach by using an extra trees classifier on estimated pedestrian poses.

The model for distracted pedestrian avoidance has four distinct stages to predict such an occurrence in the vehicle

path. The preprocessing stage passes the image through an object detector that only keeps the pixels that belong to objects recognised as people. Pose keypoint locations are obtained using a pose prediction algorithm and only poses above the horizon are kept [30]. These pixel locations of pose keypoints are processed by an extra trees classifier to predict if pedestrians are on their mobile phone. The preprocessing introduces a trade-off of greater recall for lower precision.

Using the proposed model an MCC (Matthews Correlation Coefficient) score of 0.721 was achieved on a total of 2,800 pose classifications.

E. Verification of Software

Additionally, the SOTAI regime will be realised through the application of the following three mechanisms:

- 1) **Model validation through simulated testing:** Simulation systems have long been used by major automotive manufacturers for developing and verifying advanced driver assistance systems [31]. These systems are being extended commercially and a number of open source simulation platforms suitable for developing autonomous vehicles have been developed, such as CARLA [32]. The degree of realism, especially in regard to sensor simulation, demonstrated by these simulation systems allows them to serve as the basis of a dynamic test suite for a variety of ML algorithms.
- 2) **Model supervision through diverse architectures:** Due to the size of the input space of ML algorithms used in autonomous vehicles, it is infeasible to test these algorithms on even a small fraction of their possible inputs [33]. In order to reduce the possibility of erroneous outputs from ML algorithms causing unsafe behaviours, we propose multiple algorithms, taking identical inputs and generating outputs in the same format in parallel. Each algorithm should use a substantially different architecture or, if this is not feasible, a substantially different training dataset. Through comparison of the outputs of multiple algorithms, outlying results can be removed.
- 3) **Model supervision through hard-coded constraints:** As a final level of supervision, hard-coded constraints will be developed on the outputs of ML algorithms, particularly those that generate vehicle controls. For example, the maximum steering angle and the maximum rate of change in steering angle would be limited based on the vehicle's current speed and prior physical modelling to prevent rollovers.

It is proposed that mechanisms 2 and 3 be integrated into the system design outlined in Section IV through the implementation of a series of virtual safety supervisor nodes in ROS. Each set of redundant ML algorithms will pass their outputs into a supervisor which manages the removal of outliers and selection or generation of the result. An additional supervisor placed between the navigation controller and drive-by-wire controller, enforces all supplied hard-coded constraints. These software supervisors act upon the supervisory hardware, completing our architecture.

VI. EVALUATION AND CONCLUSION

The approach presented here for design of safety systems for autonomous vehicles draws upon industrial practices and promotes the development of redundant, high integrity safeguards which are assured to address the possible hazards associated with complex systems which normally or necessarily operate under automatic control. The development of electronic control and safety systems for intelligent autonomous vehicles requires extension beyond currently defined industry standards and must account for the use of algorithms, including those defined by machine learning, by definition of a HARA process and system architecture which can identify hazards and provide means to implement safeguards for the risks associated with that intelligence.

To date, this approach has ensured the safety of the experimental vehicles utilised in our projects, however, we continue to assess and build upon the implemented safeguard as the level of autonomous functionality increases. The approach has proven fit for purpose in the research and development context, satisfying insurance and safety officers and will deliver a final product with considered and documented safety functionality.

REFERENCES

- [1] T. Drage, J. Kalinowski, and T. Bräunl, "Integration of drive-by-wire with navigation control for a driverless electric race car," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 4, pp. 23–33, 2014.
- [2] SAE J3016C, "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," SAE International, Warrendale, US, Standard, Apr. 2021.
- [3] P. Feth, R. Adler, T. Fukuda, T. Ishigooka, S. Otsuka, D. Schneider, D. Uecker, and K. Yoshimura, "Multi-aspect safety engineering for highly automated driving," in *Computer Safety, Reliability, and Security*, B. Gallina, A. Skavhaug, and F. Bitsch, Eds. Cham: Springer International Publishing, 2018, pp. 59–72.
- [4] V. V. Dixit, S. Chand, and D. J. Nair, "Autonomous vehicles: Disengagements, accidents and reaction times," *PLOS ONE*, vol. 11, no. 12, pp. 1–14, Dec. 2016.
- [5] ISO 26262-1:2018, "Road vehicles — functional safety," International Organization for Standardization, Geneva, CH, Standard, Dec. 2018.
- [6] IEC 61508-1:2010, "Functional safety of electrical/electronic/programmable electronic safety-related systems," International Electrotechnical Commission, Geneva, CH, Standard, Mar. 2010.
- [7] ISO/PAS 21448:2019, "Road vehicles — safety of the intended functionality," International Organization for Standardization, Geneva, CH, Standard, June 2020.
- [8] P. Koopman and M. Wagner, "Autonomous vehicle safety: An interdisciplinary challenge," *IEEE Intell. Transp. Syst. Mag.*, vol. 9, no. 1, pp. 90–96, 2017.
- [9] O. M. Kirovskii and V. A. Gorelov, "Driver assistance systems: analysis, tests and the safety case. ISO 26262 and ISO PAS 21448," *IOP Conference Series: Materials Science and Engineering*, vol. 534, pp. 12–19, June 2019.
- [10] J. D. Miller, *Automotive System Safety: critical considerations for engineering and effective management*. John Wiley & Sons, 2019.
- [11] D. W. Seward, F. W. Margrave, I. Sommerville, and G. Kotony, "Safe systems for mobile robots the safe-sam project," in *Achievement and Assurance of Safety*, F. Redmill and T. Anderson, Eds. London: Springer London, 1995, pp. 153–170.
- [12] R. Woodman, A. F. Winfield, C. Harper, and M. Fraser, "Building safer robots: Safety driven control," *The International Journal of Robotics Research*, vol. 31, no. 13, pp. 1603–1626, sep 2012.
- [13] H. Martin, B. Winkler, A. Leitner, A. Thaler, M. Cifrain, and D. Watznig, "Investigation of the influence of non-E/E safety measures for the ASIL determination," in *2013 39th Euromicro Conference on Software Engineering and Advanced Applications*, 2013, pp. 228–231.
- [14] F. Warg, M. Gassilewski, J. Tryggvesson, V. Izosimov, A. Werneman, and R. Johansson, "Defining autonomous functions using iterative hazard analysis and requirements refinement," in *Computer Safety, Reliability, and Security*, A. Skavhaug, J. Guiochet, E. Schoitsch, and F. Bitsch, Eds. Cham: Springer International Publishing, 2016, pp. 286–297.
- [15] D. Suo, S. Yako, M. Boesch, and K. Post, "Integrating STPA into ISO 26262 process for requirement development," in *SAE Technical Paper Series*. SAE International, Mar. 2017.
- [16] E. Galante, D. Bordalo, and M. Nobrega, "Risk assessment methodology: quantitative hazop," *Journal of Safety Engineering*, vol. 3, no. 2, pp. 31–36, 2014.
- [17] R. J. Willey, "Layer of protection analysis," *Procedia Engineer*, vol. 84, pp. 12–22, 2014, 2014 International Symposium on Safety Science and Technology.
- [18] R. Salay, R. Queiroz, and K. Czarnecki, "An analysis of ISO 26262: Machine learning and safety in automotive software," in *SAE Technical Paper Series*. SAE International, Apr. 2018.
- [19] C. B. S. T. Molina, J. R. d. Almeida, L. F. Vismari, R. I. R. Gonzalez, J. K. Naufal, and J. Camargo, "Assuring fully autonomous vehicles safety by design: The autonomous vehicle control (avc) module strategy," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, 2017, pp. 16–21.
- [20] A. Reschka, J. R. Bhmer, T. Nothdurft, P. Hecker, B. Lichte, and M. Maurer, "A surveillance and safety system based on performance criteria and functional degradation for an autonomous vehicle," in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, 2012, pp. 237–242.
- [21] K. L. Lim, T. Drage, C. Zhang, C. Brogle, W. W. L. Lai, T. Kelliher, M. Adina-Zada, and T. Bräunl, "Evolution of a reliable and extensible high-level control system for an autonomous car," *IEEE Trans. Intell. Veh.*, vol. 4, no. 3, pp. 396–405, 2019.
- [22] I. Etxeberria-Agiriano, X. Larrucea, P. Gonzalez-Nalda, M. C. Otero, and I. Calvo, "ISO26262 SEooC Compliance of a ROS Based Architecture," *Wseas Transactions On Systems*, vol. 16, 2017, publisher: World Scientific and Engineering Academy and Society.
- [23] MIL-STD-882E, "Department of defense standard practice — system safety," United States Department of Defense, Virginia, US, Standard, May 2012.
- [24] T. Drage, "Development of a navigation control system for an autonomous formula sae-electric race car," Honours Thesis, University of Western Australia, Perth WA, Australia, 2013.
- [25] EasyMile, "EasyMile EZ10 v2 Cybercar User Manual," EasyMile, User Manual Rev B04., Iter. 105, Dec. 2015.
- [26] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," Nov. 2015.
- [27] L. Reiher, B. Lampe, and L. Eckstein, "A sim2real deep learning approach for the transformation of images from multiple vehicle-mounted cameras to a semantically segmented image in birds eye view," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020, pp. 1–7.
- [28] A. P. Silvano and M. Ohlin, "Non-collision incidents on buses due to acceleration and braking manoeuvres leading to falling events among standing passengers," *Journal of Transport & Health*, vol. 14, p. 100560, 2019.
- [29] J. L. Nasar and D. Troyer, "Pedestrian injuries due to mobile phone use in public places," *Accident Analysis & Prevention*, vol. 57, pp. 91–95, 2013.
- [30] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 43, no. 1, pp. 172–186, 2021.
- [31] C. Brogle, C. Zhang, K. L. Lim, and T. Bräunl, "Hardware-in-the-loop autonomous driving simulation without real-time constraints," *IEEE Trans. Intell. Veh.*, vol. 4, no. 3, pp. 375–384, 2019.
- [32] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 1–16.
- [33] P. Domingos, "A few useful things to know about machine learning," *Commun. ACM*, vol. 55, no. 10, pp. 78–87, Oct. 2012.

CHAPTER 7

Managing Risk in the Design of Modular Systems for an Autonomous Shuttle Bus

Managing Risk in the Design of Modular Systems for an Autonomous Shuttle Bus

Thomas Drage* Kieran Quirke-Brown* Lemar Haddad* Zhihui Lai* Kai Li Lim†* Thomas Bräunl*

Abstract—This paper presents an analysis and implementation of a robust autonomous driving system for a passenger shuttle bus in shared spaces. We present results of a risk assessment for our vehicle scenario and develop a flexible architecture that integrates safety features and optimises open-source software, facilitating research and operational functionality. Identifying the Robot Operating System (ROS) framework's limitations, we incorporate our own control measures for autonomous, unsupervised operation with enhanced intelligence. The study emphasises algorithm selection based on application requirements to ensure optimal performance. We discuss system improvements, such as monitoring node implementation and localisation algorithm selection. Future work should explore transitioning to a real-time operating system (RTOS) and establishing standardised software engineering practices for consistent reliability. Our findings contribute to effective autonomous shuttle bus systems in shared spaces, promoting safer and more reliable transportation solutions.

I. INTRODUCTION

This paper presents the ongoing research on electric vehicles, vehicle automation, and autonomous driving systems at The Renewable Energy Vehicle (REV) Project, The University of Western Australia (UWA). Our current endeavour involves the development of a highly automated electric passenger shuttle bus for use as a self-driving people mover and last-mile transport solution on the university campus. This bus is designed to adapt and plan its route dynamically, with all sensory and navigation processing on-board, eliminating the need for external communication systems or remote servers. To achieve this, we have integrated and enhanced both hardware and software components to ensure reliable and safe operation [1].

The development of Level 3+ Autonomous Driving Systems (ADS) poses significant risks to both people and infrastructure due to the need for intricate software-driven electromechanical systems to ensure safe driving behaviour in complex and ever-changing environments [2]. Although autonomous vehicles are often touted as having the potential to reduce collision rates and improve traffic safety, current technology has yet to deliver on these promises [3]. As a result, significant progress is needed in the areas of safety and reliability, particularly in relation to vehicle automation systems and human intervention when necessary [4].

Historically, the automotive industry has relied on the ISO 26262 standard for ensuring the functional safety of

electronic and software systems in vehicles [5]. This standard, derived from IEC 61508 [6], has recently been supplemented by ISO/PAS 21448:2019, which provides guidelines for the design and validation of Level 1 and 2 Advanced Driver Assistance Systems (ADAS) and ADS, with scope for application to higher levels of automation [7]. The latter emphasises not only hardware failure, but also functional insufficiencies and reasonably foreseeable misuse of ADS.

In this paper, we describe a comprehensive system architecture based on fail-safe design principles, drawing from established industrial practices which target compliance with e.g. IEC 61508/61511. Our system includes control and safeguarding hardware and software components, and utilises the Robot Operating System (ROS) middleware for automation and AI implementation [8], [9]. We also discuss measures taken to ensure that our system meets the required reliability targets.

II. BACKGROUND

The REV Project acquired a pre-existing electric shuttle bus equipped with drive-by-wire hardware, but lacking any software. To repurpose this vehicle, a new compute node (Jetson AGX Xavier) and additional sensors (SBG Ellipse-D RTK-GNSS with IMU) were installed alongside the existing eight LIDARs (4 SICK, 2 Velodyne and 2 ibeo Lux). A new hardware subsystem was implemented to send drive commands to the shuttle's motor controllers. We opted for the Linux operating system and ROS robotics framework [10], selecting and modifying various ROS packages to enable autonomous driving on the university campus (Fig. 1). This autonomous shuttle bus project builds on our previous work developing an autonomous Formula-SAE car capable of detecting and navigating a racecourse marked by traffic cones [11] (Fig. 2).

It is important to note that our current focus is on developing a shuttle bus for use in pedestrian areas. Our target is to not require a human monitor to achieve the required risk reduction; allowing fully autonomous operation. Instead, we identify risks associated with component or system failure and implement appropriate safeguards, such as emergency stop procedures. However, further research is needed to validate this approach in more complex environments, such as road traffic systems. Moreover, we also consider and address potential accidents resulting from the application of safety measures themselves, which are analogous to cases of slow driver reaction to disengagement during high-speed freeway driving.

The authors are with * The REV Project, The University of Western Australia, WA, Australia thomas.drage@research.uwa.edu.au and † the Dow Centre For Sustainable Engineering Innovation, The University of Queensland, Brisbane, Qld, Australia.



Fig. 1: The nUWay autonomous shuttle bus, designed for on-campus use.



Fig. 2: The REV Formula-SAE Autonomous vehicle, a precursor to the nUWay autonomous shuttle bus project.

Our research aims to contribute to the advancement of autonomous vehicle technology by improving safety and reliability. By integrating state-of-the-art hardware and software components, we strive to develop a robust, fail-safe system architecture that meets industry standards while providing a practical solution for last-mile transportation needs on university campuses.

III. PROBLEM DESCRIPTION

The design of a systems architecture for special-purpose autonomous research vehicles which operate in a shared space presents unique challenges. Firstly, the vehicle must achieve its desired operational targets safely and reliably. Secondly, the development process must be accessible to multiple researchers and re-usable for different applications of specific deployments of such low-volume systems. REV has developed multiple such systems, which have supported research for over ten years, with evolution enabled by the increasing availability of compact, high-performance computing hardware [8], [12], [13]. As ROS has evolved as the de-facto standard for such projects [14], we leverage the ecosystem, but in doing so, introduce potential safety and reliability problems managed by the techniques described in this paper.

ROS [10] is a convenient middle-ware framework for

robot and autonomous vehicle development. It provides a publish/subscribe type messaging system with many packages, drivers and additional fast prototyping and development tools. These tools provide an excellent avenue to produce research as one can focus on a single aspect without concern for other areas. However, this system has several drawbacks that make it less reliable for real-world applications, particularly for non-technical users. Being a loosely coupled framework gives rise to a significant scope of outcomes in terms of software quality, particularly as projects become complex. Thus it becomes necessary to enforce architectural guidelines [15] or add additional software to improve robustness.

ROS provides a software framework that uses nodes to run different aspects of the navigation stack. These nodes communicate over the ROS backbone, allowing the user to focus on their primary area of research. However, ROS packages do not implement any reliability standards. In addition, many of the open-source ROS packages can cause unexpected errors when applied in different scenarios. One major issue with ROS is its typical process node starting sequence; in larger systems, manually starting up each node would be too time-consuming, so launch files are provided for an automated system start. These launch files will start each node in the file with the given parameters; however, little attention is given to the order in which nodes are started up and whether or not those nodes are started correctly. A common system failure is due to critical nodes starting out of sequence.

Another concern for ROS is the lack of system health monitoring. Nodes may fail silently in the background leading to unusual and often undesirable behaviour. For instance, the waypoint manager node may terminate abnormally, which leaves the system stranded, as no further waypoints will be added to a path. Nodes that fail silently may have other system nodes that depend on them. This failure can lead to a knock-on effect of node failures due to missing dependencies. A system designed without fault tolerance will not be able to restart or return to service once a failure has occurred.

In critical systems, such as driving algorithms for autonomous vehicles, these errors can result in catastrophic failures, from unintended property damage to injury and loss of life. Therefore we have analysed the ROS based system and implemented methods that make the system as a whole more robust and reliable. The scope includes correct configuration of the required nodes, detection of errors or stopped nodes and reporting of faults to the users.

IV. HAZARD AND RISK ANALYSIS

For risk assessment of our autonomous driving projects, we utilise a three-step process shown in Fig. 3, consisting of vehicle and ADS systemisation; then, HAZOP-style hazard identification followed by a semi-quantitative risk assessment (AV-LOPA) which is used to drive specification of safe-guarding requirements. Due to the experimental nature of our vehicle systems and iterative design approach, this process

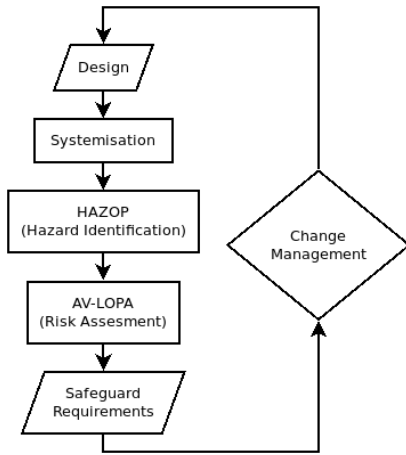


Fig. 3: HARA process flow.

is repeated as features and modifications occur, including the additions of safeguards themselves. These activities are performed as a cross-functional team in a workshop environment to provide breadths of experience and knowledge, particularly during hazard identification. Systematic application minimises the failure to identify hazards created by new functionality. This paper presents the results of the methodology described in [1] as applied to the nUWay shuttle bus and extends to the methods used to achieve the safety requirements through the automation system architecture and software framework design.

A series of five risk assessment workshops of approximately 2.5 hours each were held for the nUWay shuttle bus, examining four scenarios:

- 1) Manual (human) driving of the shuttle
- 2) Instrumented perception systems
- 3) Autonomous driving controls
- 4) Shuttle bus passenger operations

A total of 20 subsystems were examined across the scenarios and a total of 103 hazards risk assessed. Hazards were identified by the application of HAZOP guidewords defined in [1] and the University's corporate risk matrix was applied to calibrate the risk tolerance of the assessment. The outcomes are shown in Fig. 4, below, with the implication that all risk ranks must be reduced to the ranking *LOW* by application of appropriate safeguards.

Applying AV-LOPA per [1], we identified Independent Protection Layers (IPLs) of sufficient risk reduction to manage the hazards identified. These safeguards were defined to be:

- 1) **Independent:** it should not have failure modes (e.g. shared sensors) common to other safeguards used in the scenario.
- 2) **Effective:** each IPL must be able to fully mitigate the hazard.
- 3) **Validatable:** each IPL's functionality must be able to be assured and maintained.

In all cases except two, the risk was able to be reduced to *LOW*. The most commonly applied safeguard related to the

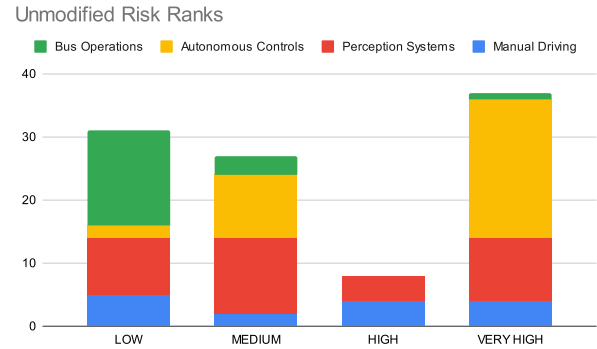


Fig. 4: Count of hazards by risk rank.

nUWay shuttle's safety LIDAR curtain system which utilises a safety PLC and four independent LIDAR sensors to prevent collision. However, in the case of 7 hazards of *VERY HIGH* risk ranking, the presence of an alert safety-driver, equipped with an emergency stop switch was required in order to rationalise the risk during autonomous driving operations. These were all related to perception system failure modes and required additional IPLs to be defined in order to eliminate the requirement for a human supervisor in the nUWay shuttle.

Additionally, two cases of *MEDIUM* risk ranking were identified for which insufficient IPLs were available and related to a false emergency stop occurring whilst at speed. The ranking of *MEDIUM* was obtained due to the high likelihood of this occurring during testing and development of the research vehicle and must be managed by a reduction of likelihood through improvement of reliability of the vehicle's systems.

V. SAFETY AND RELIABILITY REQUIREMENTS

In order to address the findings of Section IV, in particular for operation without an alert safety-driver we require to implement additional or alternative IPLs within the scope of our control and automation architecture - specifically within the high-level automation system. These IPLs are able to be independent by virtue of separate sensors (to e.g. the LIDAR safety curtain), a separate logic solver (a general purpose computer instead of a safety PLC) and separate outputs (active driving controls and alert systems). However, the challenge is in ensuring that they are effective and able to be validated, given the non-deterministic, non-realtime nature of the ROS control framework employed. If an advanced safeguard is implemented within a ROS node, and required to act as an IPL for the purposes of AV-LOPA, we must guarantee that:

- 1) The vehicle cannot proceed under automatic control unless that node is active and fault-free.
- 2) That the node is continuously monitored for operation according to its intended functionality and appropriate actions, which are not unsafe in themselves, are taken on detection of failure.

- 3) That monitoring functions cannot be disabled and have full coverage of the faults and errors which could occur within the target node.
- 4) That the node is sufficiently reliable that it performs its function without introducing further risk.
- 5) Testing is performed to assess the performance and correctness of any algorithms employed.

To achieve this within ROS we must implement the monitoring and performance evaluation functionality in conjunction with implementing the safeguards themselves. We must also provide means to validate that the ROS middleware and the operating system itself remain functional at all times, or at least that failures are detected. Given that such advanced safeguards may have multiple or unexpected failure modes, the concept of proving their effectiveness through use is expected, which can be accomplished through simulation and field trials (IX). At the same time, we must consider that nodes involved in continuous control must not introduce additional risk than was additionally assumed in the HARA through unreliability and thus apply the same constraints to their implementation; the concepts of safety and reliability in general are inextricably linked in a continuously controlled intelligent system such as this.

VI. CONTROL AND AUTOMATION ARCHITECTURE

The nUWay shuttle bus system architecture follows that of the prior Formula-SAE vehicle project, modified to suit the architecture of the commercial vehicle. In addition to the ECUs used in the drive control system, the shuttle bus features an industrial safety PLC and a computer-based high-level navigation system connected to a single CAN network. The safety PLC provided by the shuttle manufacturer handles basic sequence-control of the vehicle (e.g., interlocking doors and brakes) and provides action for fault conditions and detections from the perimeter LIDARs, which implement a simple distance-based safety curtain.

The high-level automation systems in our shuttle bus comprise two main computers; an industrial x86 PC for handling IO and core navigational functionality, and an Nvidia Jetson AGX Xavier, which provides accelerated execution of deep learning based algorithms used for interpreting sensor data. Load is distributed across the two computers based on interfacing and computational requirements to ensure maximal reliability.

As the vehicle's CAN bus specification is proprietary, for driving and steering we developed an interface system based on a TI Hercules automotive ARM microcontroller, certified for ASIL applications (Fig. 5). This development also allows us to implement additional drive-control and safety features as part of our automation system outside of the "black box" shuttle bus ECU. Additionally, it provides a hard-wired interface to the safety systems, providing redundancy and the opportunity to embed independent fault detection monitoring for the high-level systems in reliable hardware. Critical interfaces are implemented using isolated buses with loopback circuits for constant error checking, ensuring that any abnormal condition will stop the vehicle.

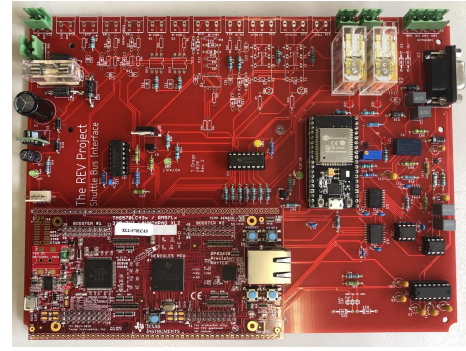


Fig. 5: REV project nUWay system interface controller.

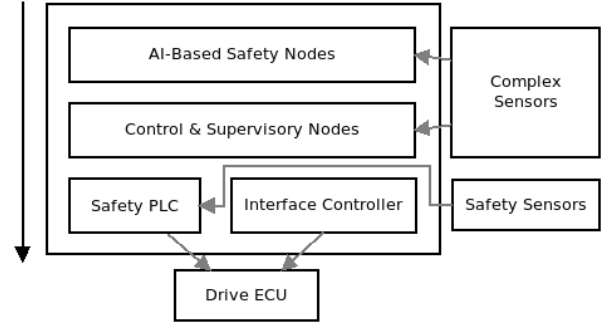


Fig. 6: nUWay Shuttle Multi-level Safety Architecture.

The nUWay shuttle bus implements several layers of safety systems shown in Fig. 6 designed to ensure that fully-autonomous operation in an environment shared with pedestrians is safe and efficient at all times. Our project has extended the safety functionality beyond low-level hazard mitigation systems and addresses the SOTIF and SOTAI regimes.

The electric shuttle bus features some built-in low-level safety features [16], which were assessed and credited as applicable in our HARA process, including:

- 1) **Obstacle detection:** Four single-beam LIDAR sensors form a detection area at a 30 cm height above ground level and secure a 2 m collision-free safety zone around the vehicle. (Fig. 7). However, this leaves the vehicle blind to obstacles lower than 30 cm.
- 2) **Emergency stops (e-stops):** Pushing one of the four in-vehicle stop buttons will immediately stop the vehicle and disarm the doors.
- 3) **Redundant braking systems:** The vehicle features multiple braking methods, including a fail-safe emergency brake.

These functions are implemented using an industrial safety PLC, which operates independently of the navigation system and interconnects with the drive-control ECUs for speed detection etc. This approach provides robustness in this application and is well suited to the specialised vehicle, however additional safeguards must be implemented at a higher level by means of software within the autonomous driving control system.

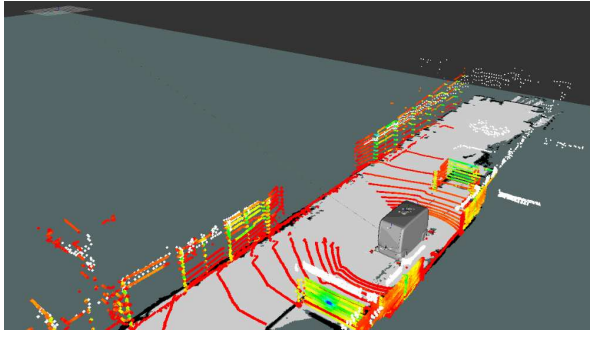


Fig. 7: LIDAR point cloud of nUWay shuttle.

VII. APPLICATION OF SOFTWARE FRAMEWORK

The Robot Operating System (ROS) is a widely adopted platform for robotics and autonomous vehicle research projects. It is utilised across various computing platforms, from embedded controllers to multicore CPUs and GPUs, and in numerous autonomous vehicles such as robots, driverless cars, drones, autonomous boats, and robot manipulators. However, its adoption within the industry is limited. Major companies such as Argo AI, Audi, Volkswagen, and Ford do not utilise ROS [17]. NASA/JPL employs ROS for development projects and proof-of-concept builds but re-implements flight systems without ROS [18]. One exception is the startup Apex.AI, which is developing an OEM-independent automotive operating system based on ROS [16].

The central concern regarding ROS is its reliability and robustness, often deemed insufficient for safety-critical systems [19]. This project's experience revealed that individual ROS software nodes, such as LIDAR sensor nodes, are prone to crashing. Consequently, additional efforts are necessary to ensure the overall vehicle system remains in a safe state, including automated detection and recovery operations to prevent collisions due to sensor information loss. Alternatives to ROS include the DDS middleware system [20], Vx-Works [21], and other real-time operating systems (RTOS). Despite the superior time and safety-critical performance of these alternatives, particularly RTOS, the modularity requirements of the framework led to the adoption of ROS, as it provides inherent modularity through its publish-subscribe architecture. It should be noted that ROS 1 lacks real-time capability, whereas ROS 2 incorporates some real-time features and more recent integration with commercial RTOS [11].

The IEEE Standard 1633-2016 defines software reliability (SR) as the probability that software will not cause a system failure for a specified time under specified conditions [22]. Unlike hardware, which is subject to wear and tear, software failures are typically due to design faults caused by human errors or oversight [23]. As such, software reliability is not affected by external conditions, nor can it be improved by running multiple instances of the same program for redundancy. Instead, design diversity, or the use of different codes performing the same task, can offer redundancy. Within our framework, the SOTAI regime, applicable to complex

ROS nodes for autonomous navigation, is realised through the implementation of three mechanisms which are detailed in [1]:

- 1) Model Validation through Simulated Testing
- 2) Model Supervision through Diverse Architectures
- 3) Model Supervision through Hard-coded Constraints

VIII. AUTONOMOUS DRIVING SOFTWARE STACK

The autonomous system was initially designed and deployed using ROS with a mixture of in-house and standard open-source packages. However, during initial testing, it was determined that more suitable packages were available in ROS 2, offering additional reliability features and the system was converted to the long-term support release of ROS 2. The following outlines the primary packages employed in the nUWay shuttle bus:

- 1) **Nav2 stack** [24]: ROS 2 offers a plugin library that provides an array of resources for hot-swapping various solutions. Moreover, it establishes a system behaviour tree with associated recovery actions, typically used to ensure sufficient progress towards the goal. If adequate progress is not made, a fallback action, such as alternate global path planning, may be performed.
- 2) **Localisation and mapping**: A crucial aspect for the shuttle bus is the capacity to build and localise on a given map. For map building, there are a limited number of packages available, with the most prominent being the SLAM toolbox [25], which utilises scan matching to generate a smooth map. The Cartographer package [26] was briefly considered but was found to be only partially implemented in ROS 2. For localisation SLAM toolbox provides a service for loading and localising a generated "pose-graph" map, however AMCL was selected as it better handles larger scale maps. AMCL requires a separate map server, which is available in Nav2.
- 3) **Global planner**: Nav2 includes several "grid-based" global planners as plugins, which use either Djikstra's or the A* algorithm to compute a path. NavFn and Smac [24] were compared to evaluate potential benefits from feasibility-based planners.
- 4) **Local planner**: The local planner selection proved critical; Nav2 provides several local planner controllers that can be "hot-swapped". The nUWay shuttle bus was tested with two primary packages: TEB (Time Elastic Band) [27] and Regulated Pure Pursuit [24], each with distinct advantages and disadvantages to reliability.

While selecting, tuning, and improving suitable open-source packages is crucial for creating a reliable driving system, the key objective of this project is to maximise reliability. This is achieved by establishing a monitor node that assesses the overall system's health and takes actions accordingly to ensure reliability. The monitor node's first aim is to initiate each node sequentially in the correct order, based on dependencies and resource requirements, to prevent node

failure and overuse of system resources. This approach also eliminates race conditions in a distributed system, such as the one on the nUWY shuttle bus, by enabling a single PC to monitor and control all nodes. On system startup, the first node is activated, and the monitor listens on the corresponding topic until node data is received, confirming successful node initiation and allowing the subsequent node to start.

Alternative system monitoring methods, such as heartbeats and watchdogs are implemented in the hardware systems and may be integrated between the monitoring node and other services in future, however, critical systems running under Nav2 are managed by a life cycle manager that handles many of these capabilities. The monitor's primary focus is on flexibility to ensure the project's scalability for future research. An external incident recorder has been developed, which activates when unexpected changes to the global path occur and records current and planned trajectories, event time, and camera data. This information feeds continuous improvement of the system.

Lastly, the monitor node establishes several monitoring sessions that captured critical information from the other nodes which is used to enhance fault tolerance. Node failures are determined by examining the current node list and listening for data on corresponding topics; allowing detection of faults in nodes that tend to fail silently. Once a failed node is detected, information about its location, failure type, and restoration methods is logged.

IX. EVALUATION OF AUTONOMOUS DRIVING

The performance of the autonomous driving system is evaluated across their respective subsections.

A. Optimising nUWY Shuttle Path Accuracy

Nine key stops across the breadth of the university campus are defined with routes between them experiencing high pedestrian traffic, especially during peak times. Groups of students further limit driving space. Fig.8 shows drives between the Law School and Student Guild stops. GPS accuracy and signal availability for Real-Time Kinematic (RTK) corrections cause slight deviations in the bus's starting position. The current system demonstrates consistent navigation using a basic path planning system.

During initial testing, unsuitable software components were replaced. Observations of student and pedestrian behaviour informed local planner selection. Initially, Navigation 2 (Nav2) implementation of Timed Elastic Band (TEB) was used for path planning due to its popularity. A performance review [28] determined that TEB and NavFn were superior in combination. However, TEB occasionally generated incoherent paths (Fig. 9a), requiring manual intervention.

Pedestrians seemed disoriented around the shuttle bus during TEB's operation due to abrupt direction changes. This led to selecting an alternative local planning algorithm. The 'regulated pure pursuit' algorithm offers a simpler alternative to TEB, functioning like a carrot on a stick. It lacks dynamic

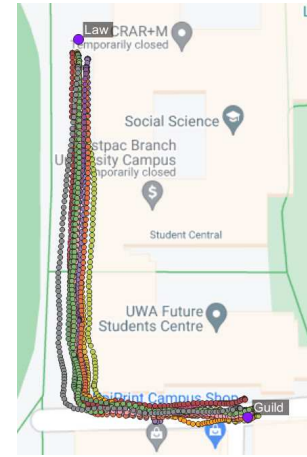


Fig. 8: Multiple recorded shuttle drives between two campus stops.

obstacle avoidance but incorporates velocity scaling based on obstacle proximity. It improved the shuttle's route following ability, and pedestrians could anticipate its path.

Initial trials of the "regulated pure pursuit" algorithm showed promising results on straight paths with minimal interference. However, it struggled on corners (Fig. 9b). Two primary solutions include using a different global planner to address feasibility concerns or tuning inflation zones for safer cornering.

Previously, the NavFn global planner was used in simple outdoor environments, while TEB handled unexpected issues. However, NavFn does not consider path feasibility, which may result in poor performance. Consequently, the Smac planner provided by Nav2 was tested but proved unsuitable due to its continuous global path adjustments.

The focus shifted to adjusting inflation zones for reliable navigation along straight paths and safe cornering. Increasing the inflation zone larger than the shuttle bus size resolved the issue. Fig. 9c demonstrates the expanded inflation zone. The light blue regions represent impassable lethal objects and a decay value is set to gradually decrease the object's severity.

B. Disengagement Events Analysis

This section evaluates the autonomous driving system performance before implementing the monitor node. We used a black-box testing approach due to the codebase's complexity, which includes open-source packages and student contributions. We characterised the system's efficacy by the mean time between failures (MTBF) during routine operations, such as path planning and navigation. Failures were assessed by their frequency, severity, and whether a system restart was required. We identified four primary areas for optimisation to improve system reliability and performance:

- 1) Low-level motor driver communication
- 2) System initialisation
- 3) Vehicle localisation
- 4) Miscellaneous driving challenges

Over three weeks, we documented 686 system failures from 57 hours of data, resulting in an average MTBF of 4.98

minutes. Table I shows the failure categories before and after improvements.

TABLE I: Sources of failure before and after improvements.

Source	Before	After
Initialisation	15.6%	1.61%
Driving	22.45%	59.68%
Localisation	60.2%	35.48%
Low-level	1.75%	3.23%

Low-level failures can result from unsuccessful message exchanges between the interface board and PLC software. Initialisation failures occur when software nodes start, while driving failures relate to local planner software issues. Localisation failures arise when the system cannot load or accurately localise within a map. These categories were further classified by severity level: low, medium, or high. Low-severity failures can be resolved by non-technical users, while medium-severity failures need technical users to restart specific nodes. High-severity failures require a complete system restart or full power cycle of the shuttle bus.

Our focus was on minimising medium and high-severity failures in the nUWay system, designed for operation by non-technical personnel. We addressed the most common issues in localisation and initialisation phases. Low-level communication failures were due to lost or delayed data packets. Enhancing load distribution and incorporating optimised data timeouts and fault recovery logic improved reliability, offering safety advantages.

A map of the shuttle's environment was created using the SLAM toolbox package, which provided a 2D map using localisation and LIDAR sensors. Despite the shuttle bus being equipped with high-quality GPS with RTK correction, a SLAM-based solution was found to be more suitable due to drift caused by nearby buildings. ROS 2 offered two packages for this purpose: SLAM Toolbox and AMCL.

Initially, SLAM Toolbox was chosen but proved inadequate for our map sizes, causing service crashes during map loading. Table II compares map loading success rates using SLAM Toolbox and AMCL. SLAM Toolbox had a 25% success rate, often needing multiple software stack restarts. In contrast, AMCL was more reliable, with a 97% success

TABLE II: Operations performed by SLAM Toolbox vs. AMCL.

Operation	SLAM Toolbox		AMCL	
	Failure	Success	Failure	Success
Map load (per hour)	5.727	2.009	0.098	3.610
Pose estimations (per hour)	1.004	2.132	0.293	6.244
Mean time between failure (minutes)	8.24		55.91	

rate and no performance issues due to smaller map sizes.

Table II shows SLAM Toolbox's poor localisation maintenance within a map, with an MTBF of 8.24 minutes. AMCL performed better, achieving an MTBF of nearly an hour. SLAM Toolbox failed to maintain localisation even with a smaller map.

C. Monitoring Node Implementation Improvements

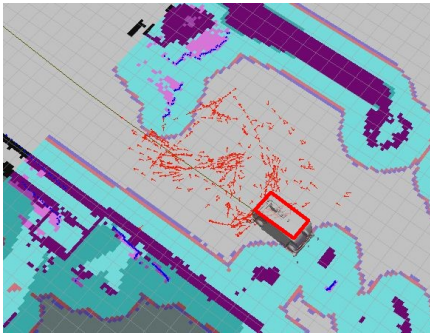
Integrating the monitoring node into the ROS 2 framework significantly enhanced the nUWay autonomous shuttle bus system performance. The startup issue with safety nodes launching before LIDAR drivers was resolved. A regulated startup sequence now ensures device drivers start and operate before activating dependent safety nodes.

TABLE III: Comparison of failure severity levels pre- and post-monitor node implementation.

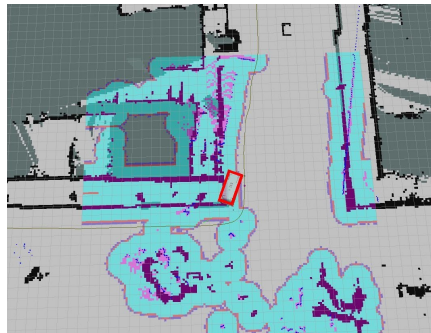
Severity	Pre-Implementation	Post-Implementation
Low	21.7%	80.7%
Medium	76.5%	16.1%
High	1.8%	3.2%

TABLE IV: MTBF comparison for severity levels and failure sources (in minutes).

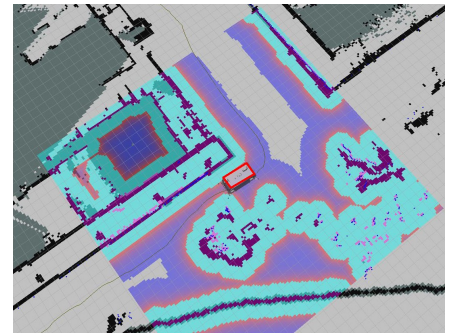
Category	Level	Initial	Final
Severity	Low	23	25
	Medium	7	123
	High	284	615
Failure Source	Low-level	284	615
	Launch	32	1230
	Localisation	8	56
	Driving	22	33



(a) Incoherent TEB paths



(b) Corner cut by regulated pure pursuit



(c) Improved by increased inflation zone

Fig. 9: Issues encountered during local planner selection and tuning.

Table III compares failure severity levels before and after monitoring node implementation, showing a significant reduction in medium severity failures. Table IV compares the MTBF for severity levels and failure categories, indicating improved system reliability and reduced frequency of high and medium severity consequences. Launch, localisation, and low-level failure probabilities decreased, leading to better overall system performance and stability primarily due to the monitoring node and better package selection. Hardware improvements are in progress to reduce low-level failures, including false positive prevention, watchdog implementation, and fault-tolerant recovery measures.

X. CONCLUSION

This study has presented a comprehensive analysis and implementation of an increasingly robust and dependable automation system for a passenger shuttle bus operating within a shared space environment. Central to our investigation was the development of a flexible systems architecture, which allowed for the seamless integration of safety features with varying degrees of complexity, as well as the optimisation of open-source software in a modular manner to facilitate the advancement of both research and operational functionality. We ascertained that the existing ROS framework does not inherently provide the necessary mechanisms to guarantee the fulfilment of our safety and reliability criteria. Consequently, we have successfully devised and incorporated our own control measures to progressively work towards our ultimate objective of achieving a fully autonomous, unsupervised operation with enhanced intelligence embedded within the vehicle. Given the promising developments in ROS 2, we anticipate utilising the emerging features and mechanisms within this distribution to augment and refine the reliability of our systems. As we continue to advance our research, we recommend that future work should focus on evaluating the feasibility of transitioning to a RTOS and establishing standardised software engineering practices. Selection of algorithms should be guided by application and not by availability of functionality within a chosen framework to ensure the best performance. These measures will ensure that safety is maintained through consistent reliability as the project expands, further solidifying the effectiveness of our proposed automation system for passenger shuttle buses in shared spaces.

REFERENCES

- [1] T. Drage, K. L. Lim, J. E. H. Koh, D. Gregory, C. Brogle, and T. Braunl, "Integrated modular safety system design for intelligent autonomous vehicles," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, jul 2021.
- [2] P. Feth, R. Adler, T. Fukuda, T. Ishigooka, S. Otsuka, D. Schneider, D. Uecker, and K. Yoshimura, "Multi-aspect safety engineering for highly automated driving," in *Developments in Language Theory*. Porto: Springer International Publishing, 2018, pp. 59–72.
- [3] V. V. Dixit, S. Chand, and D. J. Nair, "Autonomous vehicles: Disengagements, accidents and reaction times," *PLOS ONE*, vol. 11, no. 12, p. e0168054, dec 2016.
- [4] S. S. Banerjee, S. Jha, J. Cyriac, Z. T. Kalbarczyk, and R. K. Iyer, "Hands off the wheel in autonomous vehicles?: A systems perspective on over a million miles of field data," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, jun 2018.
- [5] "Road vehicles — Functional safety," International Organization for Standardization, Geneva, CH, Standard, Aug. 2018.
- [6] "Functional safety of electrical/electronic/programmable electronic safety-related systems," IEC, Geneva, CH, Standard, Apr. 2018.
- [7] "Road vehicles — Safety of the intended functionality," International Organization for Standardization, Geneva, CH, Standard, Jan. 2019.
- [8] K. L. Lim, T. Drage, C. Zhang, C. Brogle, W. W. L. Lai, T. Kelliher, M. Adina-Zada, and T. Braunl, "Evolution of a reliable and extensible high-level control system for an autonomous car," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 3, pp. 396–405, sep 2019.
- [9] X. Larrucea, P. González-Nalda, I. Etxeberria-Agiriano, M. C. Otero, and I. Calvo, "Analyzing a ROS based architecture for its cross reuse in ISO26262 settings," in *Communications in Computer and Information Science*. Cham: Springer International Publishing, 2018, pp. 167–180.
- [10] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2, Kobe, Japan, 2009, p. 5.
- [11] ROS Tutorial, "Real-time programming in ROS 2," May 2022. [Online]. Available: <https://docs.ros.org/en/foxy/Tutorials/Real-Time-Programming.html>
- [12] K. L. Lim, T. Drage, R. Podolski, G. Meyer-Lee, S. Evans-Thompson, J. Y.-T. Lin, G. Channon, M. Poole, and T. Braunl, "A modular software framework for autonomous vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, jun 2018.
- [13] T. H. Drage, "Development of a Navigation Control System for an Autonomous Formula SAE-Electric Race Car," Master's thesis, The University of Western Australia, 2013. [Online]. Available: <http://robotics.ee.uwa.edu.au/theses/2013-REV-Navigation-Drage.pdf>
- [14] A. Araujo, D. Portugal, M. S. Couceiro, and R. P. Rocha, "Integrating arduino-based educational mobile robots in ROS," in *2013 13th International Conference on Autonomous Robot Systems*, apr 2013.
- [15] I. Malavolta, G. A. Lewis, B. Schmerl, P. Lago, and D. Garlan, "Mining guidelines for architecting robotics software," *Journal of Systems and Software*, vol. 178, p. 110969, aug 2021.
- [16] J. Becker, "Das neue auto-os und die robotik," Interview, May 2022. [Online]. Available: <https://intellicar.de/podcast/das-neue-auto-os-und-die-robotik/>
- [17] A. Boeing, Private Communication, Munich, May 2022.
- [18] I. Nesnas, Private Communication, Pasadena, Jan. 2016.
- [19] ROS Answers, "What are technical reasons for criticisms of ROS's Reliability/Robustness/Safety?" May 2022. [Online]. Available: <https://answers.ros.org/question/317435/what-are-technical-reasons-for-criticisms-of-ross-reliabilityrobustnesssafety/>
- [20] DDS Foundation, "What is DDS?" May 2022. [Online]. Available: <https://www.dds-foundation.org/what-is-dds-3/>
- [21] Wind River Systems, "VxWorks - The Leading RTOS for the Intelligent Edge," May 2022. [Online]. Available: <https://www.windriver.com/products/vxworks>
- [22] "IEEE recommended practice on software reliability," *IEEE Std 1633-2016 (Revision of IEEE Std 1633-2008)*, pp. 1–261, 2017.
- [23] M. R. Lyu, *Handbook of software reliability engineering*. Piscataway, NJ: IEEE Computer Society Press, 1996.
- [24] S. Macenski, F. Martin, R. White, and J. G. Clavero, "The marathon 2: A navigation system," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, oct 2020.
- [25] S. Macenski and I. Jambrecic, "Slam toolbox: Slam for the dynamic world," *Journal of Open Source Software*, vol. 6, no. 61, p. 2783, 2021.
- [26] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, may 2016.
- [27] C. Roesmann, W. Feiten, T. Woesch, F. Hoffmann, and T. Bertram, "Trajectory modification considering dynamic constraints of autonomous robots," in *ROBOTIK 2012; 7th German Conference on Robotics*, 2012, pp. 1–6.
- [28] A. Filotheou, E. Tzardoulis, A. Dimitriou, A. Symeonidis, and L. Petrou, "Quantitative and qualitative evaluation of ROS-enabled local and global planners in 2d static environments," *Journal of Intelligent & Robotic Systems*, vol. 98, no. 3-4, pp. 567–601, oct 2019.

Conclusion

Summary

This thesis presented a collection of works which describe the development of automation systems for two autonomous vehicles. The vehicles were developed as extensible platforms for safe research into localisation, navigation, machine learning and systems engineering, culminating in the application of an autonomous shuttle bus to on-campus people movement. The research presents an evolution of the systems architecture, driven by improvements in available technology and the desire to increase the capability, reliability and accessibility of the vehicles, with progressively more sophisticated techniques successfully applied. New approaches were proposed which enable the use of open-source middleware while managing safety risk introduced by the automation.

The first chapter described the development of the *Formula-SAE Autonomous* vehicle, one of the world's first autonomous electric vehicles of its class. This initial research platform utilised a custom-built software and hardware framework and relatively modest computational hardware. Chapter Two has described the creation of a key algorithm, for road-keeping, which offers both control and safety advantages to the vehicle utilising a sophisticated sensor and efficient processing. The algorithm presented is novel in its ability to perform well in spite of operation on a poorly defined road or pedestrian path. This approach is contrasted with machine learning based semantic segmentation in Chapter Three and which also proved to offer good performance as well as wider advantages for autonomous navigation.

Chapters Four and Five examined the development of the software framework and transition from a custom monolithic application, to a modular application design and finally to adoption of the ROS middleware [1]. This process was driven by desire for utilisation of free and open-source components, improving accessibility of the platform for researchers to test components and the implementation of more

advanced, high-level algorithms, including those which leverage machine learning. The transition was enabled by the improvement in capability of embedded computer systems, specifically low-power multi-processor systems with the opportunity for hardware accelerated computer vision and machine learning applications. Throughout the project, all sensing and control was performed on-board without resorting to full-sized desktop or server systems. The move to ROS was found to have benefits and limitations and so components of the monolithic architecture were retained, as ROS modules themselves.

The final two chapters detailed the application of our architecture to an electric shuttle bus, *nUWay*, demonstrating applicability in advancing electromobility in a useful fashion as an on-campus people mover. To facilitate this application, a risk assessment process was formally defined and a method for ensuring that safeguards sufficiently mitigate the risks. The examination focussed on the goal of eliminating a human backup to the autonomous driving system for the purpose of creating a truly driverless intelligent vehicle. A systems architecture which allowed us to integrate with the shuttle bus systems and achieve our goal of implementing navigation functionality as well as safeguards was presented. Finally, the benefits of this architecture were evaluated in the context of the development and testing of the shuttle bus.

Evolution of Systems Architecture

The hardware architecture evolved in this project in pursuit of improved reliability and greater (sensing) functionality; it was enabled by the engineering of more robust interfaces, the procurement of more advanced sensors and the improvement of the performance of embedded microprocessors. Embedded systems replaced safety interlocks implemented in analogue circuitry and multiple low-integrity micro-controller systems on the *Formula-SAE Autonomous* vehicle were replaced with the use of an automotive grade safety micro-controller in conjunction with industrial safety PLC and drive ECU on *nUWay*. We procured more advanced GPS/INS systems (with integrated high-rate GPS/IMU fusion), moved from a single LIDAR to a suite offering full coverage at multiple distances and added a high quality stereo-vision system. During the period of this research, we moved from an Intel Atom processor with some 47 million transistors [2] to the NVIDIA AGX Xavier with 9 billion [3]. Additionally, new micro-architectures are designed specifically to target the AI/ML applications required for advanced systems,

especially autonomous driving and so allowed our project realise performance improvements from the ability to process and model data from multiple sensors, which was not possible at the start. The result of this additional capability was the ability to move our research from the simplistic race-track application of the *Formula-SAE Autonomous* vehicle to *nUWAg* which operates in a mixed environment without pre-defined path.

The software architecture evolved with the motivation of improving the ability to integrate and evaluate new algorithms and capabilities as software modules whilst constrained by available computational power. The original monolithic architecture was performant but segregating features for testing outside of the vehicle system was difficult. The move to ROS was initially resisted, in favour of modularising the original system which allowed independent off-line testing of the individual modules. However, the availability of sophisticated and documented API and open-source implementations of useful algorithms drove the eventual transition. By this time, improved hardware removed the constraints previously experienced and whilst some benefits were realised, the node based architecture required additional work to ensure reliability and dependability of the overall system.

Findings

The key findings of this research span four primary areas; applicability of LIDAR sensing, the advantages and disadvantages of the ROS middleware, the use of diversity within a systems architecture to achieve safety and the need for extension of safety standards to suit highly intelligent vehicles.

Applicability of LIDAR Sensing

In this project LIDAR sensing was used for; object detection and tracking, road edge detection, navigation within a cone-marked track, to detect nearby obstacles and prevent collisions, and for localisation in the form of SLAM. A variety of LIDAR sensors were employed, including single scan, long-distance multi-layer units and 360 degree 3D sensors all of which proved to be excellent in performing their required function. Early in the project, when targeting the available embedded computers, LIDAR was preferred as high rate data could be processed with lower computational load, however, the platforms available now (e.g. NVIDIA Jetson AGX) are well suited to video processing and have obviated

this factor. In general, it was found to achieve our goals multiple LIDARs of different types were required but that they were readily integrated and the data processed easily with the available hardware and software frameworks, both open-source and custom developed.

Comparison with computer vision methods was performed for all of these applications and it was found that the performance was comparable, but that false detections were an issue with semantic segmentation and that specialised hardware and ultimately distributed computing was required to be able to handle the load of processing vision data. Computer vision techniques based on machine learning have improved in recent years and future work should include optimisation of the image capture system and a more robust evaluation of the relative performance of segmentation methods. We conclude however that computer vision does offer some specific advantages in being able to develop increasingly sophisticated models of vehicle's environment (for example in monitoring pedestrians) and therefore recommend that LIDAR and computer vision are best employed as complementary techniques.

Advantages and Disadvantages of ROS Middleware

Whilst computationally efficient, the use of a monolithic, custom developed control software was found to limit extensibility and hinder collaboration on the development of our autonomous driving platform, and by extension research into new methods for sensing and navigation. Hybrid models, incorporating significant elements of the monolithic system as ROS nodes were found to be effective, however, the availability of open-source navigation components drove the full adoption of the ROS middleware and an architecture to suit. The benefits of standardisation of interfaces and ease of access and support for researchers were realised. However, it was found that ROS, due to its node-based loosely coupled architecture and previous applications in field robotics, was not optimally suited to the needs of a large road vehicle. In particular, the structure of the framework does not provide sufficient means for ensuring that nodes which are co-dependent are functioning correctly or for detection and recovery of indeterminate states. It is difficult to guarantee inter-node communication and timing in a non-real-time system. Thus, significant work was required to bolster and adapt the implementation to ensure safety and reliability of the system.

The use of free and open-source components for localisation and navigation was found to bring some benefits and some risks. Many ROS packages are not as fully developed as hoped and often do not generalise well to vehicle types other than those for which they were specifically developed. It was commonly found that they are not particularly robust and that compatibility between versions and packages is not guaranteed. The claim that re-use of an open-source package will offer improved functionality or decreased development time was therefore not proven, however future developments in ROS do appear promising.

The Use of Diversity to Achieve Safety

A theme developed across both the architecture of the *Formula-SAE Autonomous* vehicle and the *nUW* shuttle bus, was the use of diverse or redundant sensors and algorithms to ensure safety of the vehicle's operation. Both vehicle's architectures featured custom-built hardware which provided means to implement hardware monitoring and safety logic separate to the main (computer-based) drive controller. This approach was found to be effective when considering basic low level safeguards (e.g. implemented purely in hardware) and extended to more a more sophisticated implementation consisting of safeguards across multiple systems with diverse sensors. The approach is particularly useful in the case of our research platforms, as it is possible to identify which components are required for safety and which may be subject to testing and development without increasing risk excessively. Significantly, it was found through analysis of the safeguards present that this approach, incorporating both low level and sophisticated software safeguards, is required in order to increase the level of automation of the *nUW* shuttle and eliminate any reliance on a human backup operator. For example, in order to prevent collisions with pedestrians in a supervision free driving mode, we identified the need to augment the shuttle's LIDAR safety curtain system with a preventative layer utilising computer vision to identify paths with lower risk. To enable this, we defined and refined an architecture which supports implementation of redundant and independent functions, providing the ability to create a multi-level safety solution.

Safety Standards and Methods for Highly Intelligent Vehicles

Standards applicable to the development of safety related electronic components in road vehicles such as ISO 26262 [4] and those which focus on ADAS such as ISO/PAS 21448 [5] were examined as applicable to a highly intelligent

autonomous vehicle. It was found that further extension is required in order to manage the safe development of systems for vehicles, which by nature of their use cases, require sophisticated algorithms, including machine learning methods for sensing and navigation. This concept was defined as Safety of the Artificial Intelligence (SOTAI). Furthermore, a framework was developed where additional such algorithms may be implemented and assessed as safeguards themselves in order to create layers of protection through diversity, isolation and redundancy of components. It was noted that very recently, updates extending the scope of standards (e.g. ISO/PAS 21448:2022) have begun to address a wider range of issues emerging from development and commercialisation of SAE Level 3+ systems.

Future Work

The prospects for the autonomous driving platforms of the *REV Project* are extremely promising and future work is expected to encompass the realms of sensing, localisation, navigation, safety and reliability. The performance of the vehicle to date is not yet sufficient for totally unmanned operation – to achieve this further optimisation of the implementation including potential replacement of some of the open-source components with more tailored implementations is necessary. In particular, the local planner utilised within ROS on the *nUWay* shuttle resulted in numerous failures during driving and is a key candidate for further work to evaluate the modification of the module or replacement with a more reliable bespoke implementation.

In the current state of automation it was identified through risk assessment that false-triggering of the safety curtain system (e.g. due to rain) resulting in abrupt stopping of the *nUWay* shuttle could present an unacceptable risk to standing passengers. Testing should be carried out to investigate whether this is possible and if it is considered too likely, it should be mitigated through an engineering effort. Further safeguards are recommended to be implemented including a redundant low-level check of speed control discrepancy and the operational implementation of the pedestrian collision avoidance system using computer vision before unattended operation is realised. Reliability analysis should be evergreen in terms of improving the system uptime and it is expected that as development progresses the hazard and risk assessment will be iterated in order to take on changes to the system and to the understanding of the use case based on the field trials currently in progress.

It is highly recommended that future work includes a comparative assessment of a navigation control system implemented within an RTOS (real-time operating system) framework, potentially utilising emerging features and integrations in ROS, or without ROS. Of note is the Space ROS [6] project which seeks to adapt ROS to achieve the reliability and assurance features required for space robotics and which may introduce innovations useful in automotive applications. It is expected that in many cases ROS represents a stepping stone towards identifying the scope and components of a system which can then be re-implemented using a less flexible but more reliable framework. The further evolution of the system should focus on the ability to test and validate components, their failure modes and actions on detected failures.

In recent years the progress of the automotive industry towards vehicle automation and highly sophisticated ADAS systems has accelerated, although the pathway towards full driverless automation still contains ethical and legal roadblocks. Future work in the industry should focus on extension of standardisation to ensure that the systems which are developed are both well designed but also able to be validated independently. Machine learning based technologies will be a part of the future of road vehicle automation and such implementations must be approached with the utmost care in order to not introduce excessive risk or systems which rely entirely on components which cannot ever be fully tested. The development of autonomous vehicles heralds a great benefit to society, but the commercialisation requires manufacturers and regulators to put the goal of achieving a verily safer road transport system ahead of all others.



References

- [1] Open Source Robotics Foundation, “ROS/Introduction - ROS Wiki.” [On-line]. Available: <http://wiki.ros.org/ROS/Introduction>. Accessed on: May 29, 2019.
- [2] Intel Corporation, “Intel Atom® Processor N270” [On-line]. Available: <https://www.intel.com/content/www/us/en/products/sku/36331/intel-atom-processor-n270-512k-cache-1-60-ghz-533-mhz-fsb/specifications.html>. Accessed on: October 1, 2023.
- [3] NVIDIA Corporation, “NVIDIA Jetson AGX Xavier Delivers 32 TeraOps for New Era of AI in Robotics” [On-line]. Available: <https://developer.nvidia.com/blog/nvidia-jetson-agx-xavier-32-teraops-ai-robotics/>. Accessed on: October 1, 2023.
- [4] ISO 26262-1:2018, “Road vehicles — functional safety,” International Organization for Standardization, Geneva, CH, Standard, Dec. 2018.
- [5] ISO/PAS 21448:2019, “Road vehicles — safety of the intended functionality,” International Organization for Standardization, Geneva, CH, Standard, June 2020.
- [6] Open Source Robotics Foundation, “Space ROS – Frequently Asked Questions” [On-line]. Available: <https://space.ros.org/pages/faq.html>. Accessed on: April 22, 2023.