Student ID:	218356779
Student Name:	MCINTYRE, GREGORY STEPHEN
Campus:	☐ Burwood ☐ Waterfront ☐ Waurn Ponds ☐ Warrnambool ☒ Cloud

Assignment Title:	SIT310: Robotics Application Development: Final Report		
Due Date:	3/6/2019	**Assessment Item:**	Report
Course Code/Name:	ROBOTIC APPLICATION DEVELOPMENT		
Unit Code/Name:	SIT310	**Unit Chair / Campus Coordinator:**	Kevin Lee
Practical Group: (if applicable)	N/A		

PLAGIARISM AND COLLUSION

Plagiarism occurs when a student passes off as the students own work, or copies without acknowledgement as to its authorship, the work of another person. Collusion occurs when a student obtains the agreement of another person for a fraudulent purpose with the intent of obtaining an advantage in submitting an assignment or other work. Work submitted may be reproduced and/or communicated for the purpose of detecting plagiarism and collusion.

DECLARATION

I certify that the attached work is entirely my own (or where submitted to meet the requirements of an approved group assignment, is the work of the group), except where work quoted or paraphrased is acknowledged in the text. I also certify that it has not been previously submitted for assessment in this or any other unit or course unless permissions for this has been granted by the Unit Chair of this unit. I agree that Deakin University may make and retain copies of this work for the purposes of marking and review, and may submit this work to an external plagiarism-detection service who may retain a copy for future plagiarism detection but will not release it or use it for any other purpose.

Signed:	GREGORY STEPHEN MCINTYRE	**Date:**	29/5/2019

An assignment will not be accepted for assessment if the declaration appearing above has not been signed by the author. If submitting electronically, print your full name in place of a signature.

COMMENTS					
Mark Awarded:		**Assessor's Signature:**		**Date:**	

Marking Grid - Report - (30% of Unit)

Criteria	Basic (1-2)	Intermediate (30)	Advanced (4)	Complete (5)	Marks Awarded
Problem Description	Problem is not well defined or description lacks significant detail needed to understand problem features or why it is a problem	Problem description lacks detail and demonstrates insufficient understanding of problem features	Description conveys major features of problem and relates relevant dimensions that make the problem hard to solve	Description presents cohesive and thorough understanding of the problem and why it is difficult to solve given current knowledge and capabilities in computing and/or engineering	/5
Requirements of smart-car	List of unstructured system requirements	Functional and none-functional (quality) requirements listed. Good attempts relate these to high-level system requirements.	Ranked requirements based on an estimation of effort or time needed.	Relationships between requirements clear including risks of not completing features.	/5
Overall ROS design and ROS Process graph	Simple description of how the base system can be built using ROS. Simple ROS process graph.	Identified and justified ROS technologies to use. Annotated ROS graph.	Description conveys a clear design of how to implement the solution in ROS.	Description presents cohesive and thorough understanding of how to apply ROS to a problem.	/5
Advanced Feature 1	Appropriate feature identified. Basic description of feature. Justification of the need for the feature.	Details of how it can be implemented in ROS. Figures or diagrams to illustrate the feature.	Description conveys a clear design of how to implement the solution in ROS.	Description presents cohesive and thorough understanding of how to apply ROS to this feature.	/5
Advanced Feature 2	Appropriate feature identified. Basic description of feature. Justification of the need for the feature.	Details of how it can be implemented in ROS. Figures or diagrams to illustrate the feature.	Description conveys a clear design of how to implement the solution in ROS.	Description presents cohesive and thorough understanding of how to apply ROS to this feature.	/5
Advanced Feature 3	Appropriate feature identified. Basic description of feature. Justification of the need for the feature.	Details of how it can be implemented in ROS. Figures or diagrams to illustrate the feature.	Description conveys a clear design of how to implement the solution in ROS.	Description presents cohesive and thorough understanding of how to apply ROS to this feature.	/5
Overall:					/30

Highlight what you believe you have achieved. Note that poor use of presentation and English will reduce marks.

[ROBOTICS APPLICATION AND DEVELOPMENT]

REPORT

An investigation into the problems with modern automated vehicles and an engineering approach to Navigation using line sensors, Lane Changes and Conflict Resolution.

[GREGORY STEPHEN MCINTYRE]

[218356779]

Executive Summary

Automated driving is a difficult task to achieve given our current level of computing capability and knowledge. the Society of Automotive Engineers (SAE) has broken down the minimum base requirements that a smart car needs to achieve into six categories: Level 0, no level of automation. Level 1, basic user assistance. Level 2, partial automation. Level 3, conditional automation. Level 4, high automation. Level 5, full automation. An automated vehicle must be able to complete many tasks using the DARPA model these are: sensing is the collection of raw data for further analysis, the accuracy and timeliness of data collection allow to produce better raw data. Perception is the creation of usable information from the sense data or from other sub systems, the risk of not creating accurate perception data can lead to inaccuracies, unexpected behaviours, a lack of reaction or overreaction within planning. Planning, A lack of accurate planning will greatly reduce efficiency and the effectiveness of the vehicle's operation. Control sub-system involves the operation of the vehicle itself, a lack of accurate control many have downstream effects with the perception and planning sub-systems.

The Robot Operating System (ROS) can be used to implement and manage many of the features of an automated vehicle. ROS is designed with the functional use of nodes and topics allows for a great deal of customisation without the detrimental impact of other modules.

This report reviews the implementation of three advanced features. Navigation using Line Sensors, the front sensor array comprises of five line-sensors and three proximity-sensors. This feature works by using a sensing node on the Arduino to pass line sensor topics to the perception and planning node where they modify the control topic to have the robot turn away from the white lines of the road.

The automatic changing of lanes requires the enabling of three of the proximity sensors in order detect an object in front using either the front left sensor or front right, and to detect when the object has been passed using the left sensor. The sensing node passes to the sensor topics to the perception and planning node that are multiplexed into the control node alongside the other nodes.

Conflict Resolution, this feature very simply implements an emergency stop. If an object is detected to be within very close proximity to the robot, then it will immediately halt all movement and wait for further instruction. Many post-accident reviews found that 'swerving' was too unpredictable and could be caused by false positives adding to risk. Thus, this module does not implement a swerve.

Contents

Introduction

Automated Vehicles have been a thing of imagination for a long time. In 1935, David H. Keller wrote a book called 'The Living Machine' A science fiction novel, where the driverless taxi was introduced as a point of conflict, and since then the idea has only gained in popularity, Isaac Asimov, 'Sally' 1953, Arthur C. Clark, 'Imperial Earth' 1976, and on and on until the current day, and now in 2019 the possibility of vehicles without human drivers is as close as it has ever been.

This Report contains two major parts. The first, a breakdown of the problem that automated vehicles must overcome, why it is difficult to solve given current technology and knowledge, with links to the relationships between requirements and the risks of not completing them, an explanation of the Robotic Operating System (ROS) architecture and how I can be applied to these problems. The second part presents three advanced automated driving feature discussions and provides a cohesive and through explanation on how ROS could be used to implement a solution.

The Obstacles of Automates Driving

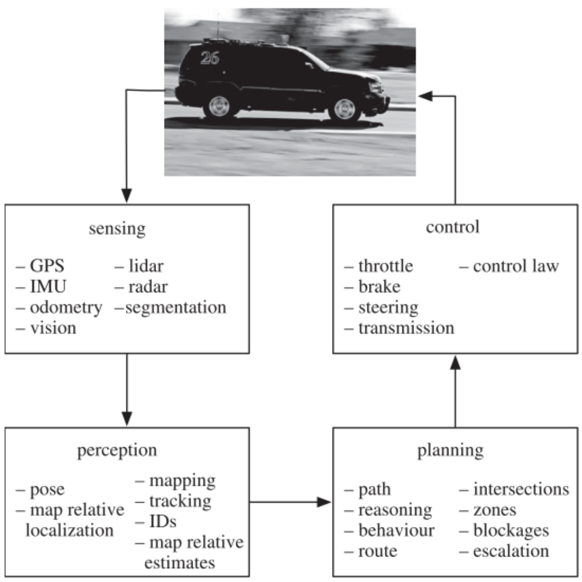
Automated driving is a difficult task to achieve given our current level of computing capability and knowledge. In this section this it will present cohesive and thorough analysis of the current state of automated driving technology including, a description of the problems of automation and the base requirements an automated vehicle must achieve, an analysis of the tasks that an automated vehicle must do, how the Robot Operating system (ROS) could be used to achieve them and a ROS process graph.

the Society of Automotive Engineers (SAE) has broken down the minimum base requirements that a smart car needs to achieve into six categories (SAE, n. d., SAE On-Road Automated Vehicle Standards Committee, 2014, Coppola and Morisio, 2016).

- Level 0, at level 0 there is no level of automation, this means that all the functions of the vehicle must be manually controlled by the vehicle operator (driver).
- Level 1, at level 1 there is basic user assistance tools available for example driver assist, lane control or parking sensors to alert the driver. Because the vehicle is able to alert the user to potential dangers it is often seen as essential to providing improved safety to the vehicle occupants (Coppola and Morisio, 2016).
- Level 2, partial automation, at this level the vehicle is capable of diving autonomy, but the diver must be able to identify and take control of the vehicle in unexpected, possibly dangerous situations. Partial automation can have many benefits such as improved safety, improved efficiency and reduction of mental load (Casner et al., 2016)
- Level 3, at this level there is conditional automation meaning, the vehicle is capable of monitoring is surrounds and informs the driver when a takeover is required. The current limitation to level 3 autonomous vehicles is the speed and quality of sensor data and the over filtering of quality data (Mössinger, 2010).
- Level 4, high automation, at this level the vehicle is capable of its own emergency management and can react to unexpected situations if the driver is unable to takeover This is a difficult task due to many factors including, weather, roads without lines, high speed situations (Barabás et al., 2017) and dealing with varying human factors such as road rage, erratic driving and panic behaviours (Saffarian et al., 2012).

- Level 5, full automation, at this level the driver is completely unnecessary. This is a difficult task for many reasons, including; coexistence with non-autonomous vehicles, the quality utilisation of existing infrastructure that is designed for humans not robots (road to vehicle communication), vehicle to vehicle communication, vehicle to internet communication (Saffarian et al., 2012, Barabás et al., 2017).

An automated vehicle must be able to complete many tasks the most recognised definitions of these tasks are from the US Defence Advanced Research Projects Agency (DARPA) challenge; sensing, perception, planning and control (Norton et al., 2017, Campbell et al., 2010). For this analysis, the DARPA model will be used due to its simplicity and lack of overlap.



(Campbell et al., 2010)

Figure 1: DARPA Automated Vehicle System Architecture

Sensing is the collection of raw data for further analysis, this could include, the use of Global Positioning Systems (GPS), Inertial Measurement Units (IMU), collection of odometry data, visual data, compass or bearing, Radio Detection and Ranging (RaDAR), Light Detection and Ranging (LiDAR), image segmentation and many more. Sensing is a slow developing part of this process due to its reliance on hardware (Campbell et al., 2010). Due to the dynamic environment, the vehicles must operate, the accuracy and timeliness of data collection allow to produce better perception data. An example of poor sensing maybe the poor use of GPS, leading to poor perception data that could result in the vehicle positioning itself poorly leading to a dangerous situation for both driver and other road users.

Perception is the creation of usable information from the sense data or from other sub systems. This includes; the perception and identification of objects in visual data, the cartesian conversion of position and pose, map relative locations and estimates, mapping from detection and ranging systems, tracking of objects (Campbell et al., 2010). The risk of not creating accurate perception data can lead to inaccuracies in planning, and unexpected behaviours, if perception data is not generated, to not generated well it can lead to a lack of reaction or overreaction within planning. A common error caused by poor perception is the miss detection of partially obstructed objects, poor machine vision can lead to the miss identification of threats that can lead to a reduced reaction by the vehicle.

Planning utilises more developer creativity for; pathfinding, reasoning, safety behaviour, behaviour at events such as intersections, speed zones, obstacles and threat escalations. A lack of accurate planning will greatly reduce efficiency and the effectiveness of the vehicle’s operation. The risks of this behaviour can be minor or sever depending on the levels of inaccuracy, poorly designed route planning may result in slow travel times, but negligent route planning could result in very dangerous vehicle behaviours.

The control sub-system involves the operation of the vehicle itself. This system would control; the throttle, the brakes, steering and transmission. A lack of accurate control many have downstream effects with the perception and planning sub-systems, if the vehicle is not braking as it should the planning system may inaccurately predict the stopping time of the vehicle and this may lead to a negative feedback loop and increased risk and possible injury.

The Robot Operating System (ROS) can be used to implement and manage many of the features of an automated vehicle. ROS is a modular set of software libraries that can be used to control a robot, its designed with the functional use of nodes and topics allows for a great deal of customisation without the detrimental impact of other modules. In Figure 2, a process graph can be seen that aligns with some of the concepts expressed so far in this report, far from being complete this diagram is merely to visualise some possible nodes and topics the system may have.

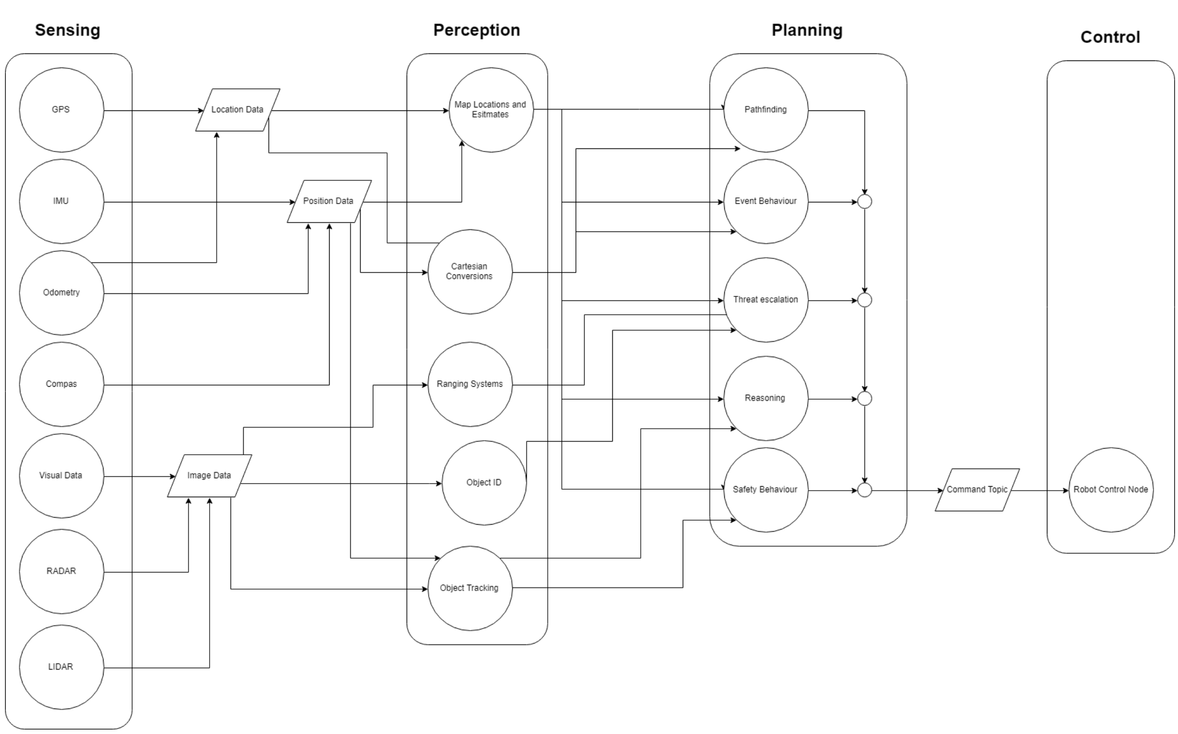


Figure 2: Possible ROS Process Graph for an Automated Vehicle

In conclusion, automated driving is a difficult task to achieve. Given our current level of computing capability and knowledge level 3 or maybe level 4 are possible but require much more investment. In this breakdown it presented a cohesive and thorough analysis of the current state of automated driving technology including, descriptions of the problems automated vehicles face and the base requirements an automated vehicle must achieve to achieve SAE recognition. It performed an analysis of the tasks that an automated vehicle must do, how the Robot Operating system (ROS) could be used to achieve them.

Navigation using Line Sensors

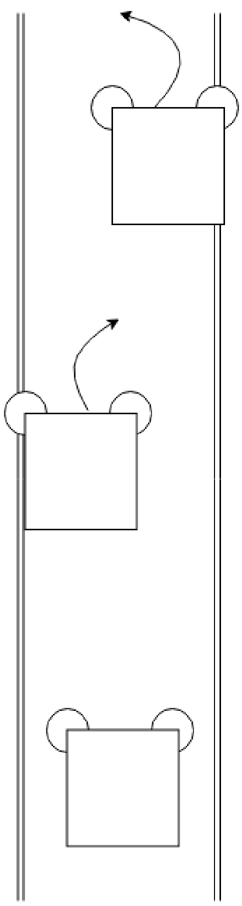


Figure 3: Line Sense

Implementing navigation using a line sensor will require the use of the front sensor array (Pololu Robotics & Electronics, n.d.). The front sensor array comprises of five line-sensors and three proximity-sensors. For the purposes of this feature, it will not be required to activate the proximity sensors at this point.

The Arduino code for the Zumo 32U4 can be found here:

https://github.com/gregorymcintyre/RoboticsApplicationDevelopment/tree/master/Task%209%2010%2011%20-%20Project/1.%20Navigation%20using%20Line%20Sensors/LineSensor

The required Scripts can be found here:

https://github.com/gregorymcintyre/RoboticsApplicationDevelopment/tree/master/Task%209%2010%2011%20-%20Project/1.%20Navigation%20using%20Line%20Sensors/scripts

To run the feature, the following commands will be required:

```
$     roscore
$     rosrun rosserial_python serial_node.py
/dev/ttyACM0
$     rosrun project zumo_move_forward.py
$     rosrun project zumo_line_detect.py
```

The following feature works by using a sensing node on the Arduino (LineSenor.ino) to pass line sensor topics to the perception and planning node (zumo_line_detect.py) where they modify the control topic to have the robot turn away from the white lines of the road.

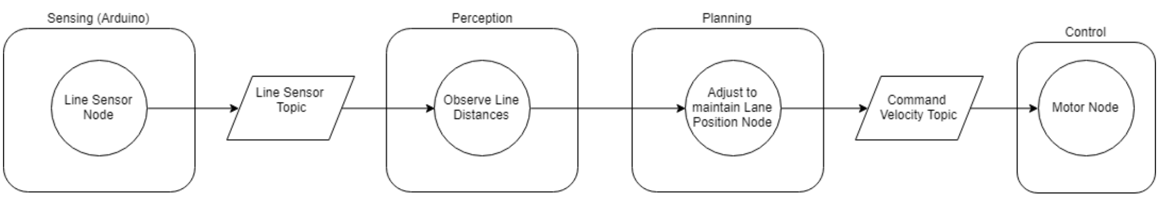


Figure 4: Navigation by Line Sense ROS Diagram

Automatic Changing of Lanes

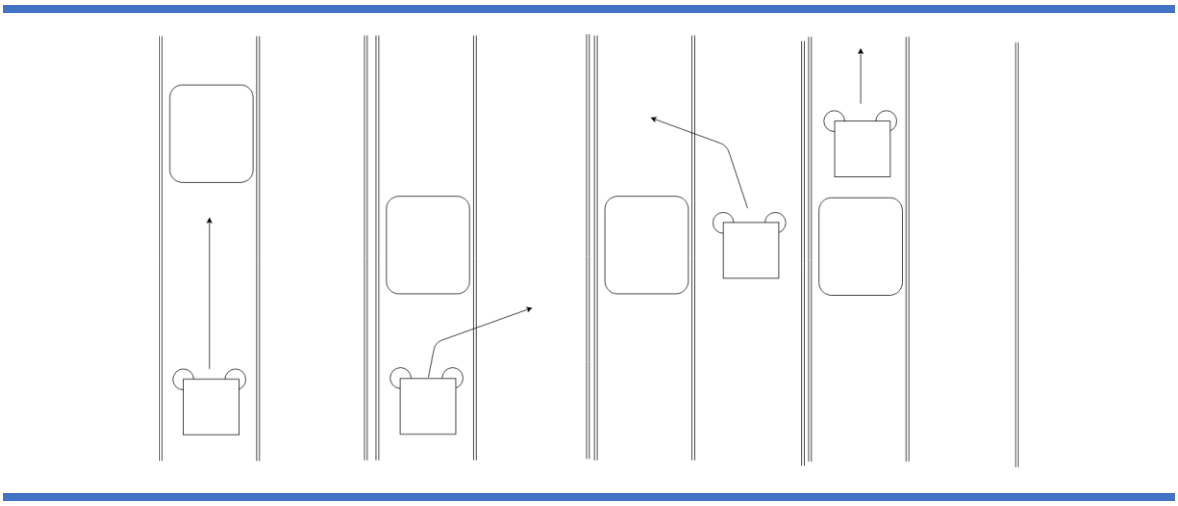


Figure 5: Lange Change

Implementing the automatic changing of lanes will require more implementation of the front sensor array implemented in the last feature. It will require the enabling of three of the proximity sensors in order detect an object in front using either the front left sensor or front right, and to detect when the object has been passed using the left sensor. The changing of lanes is designed to only overtake to the right of the vehicle as to adhere to basic Australian road rules.

The Arduino code for the Zumo 32U4 can be found here:

https://github.com/gregorymcintyre/RoboticsApplicationDevelopment/tree/master/Task%209%2010%2011%20-%20Project/2.%20Automatic%20Changing%20of%20Lanes/LaneChange/

The required Scripts can be found here:

https://github.com/gregorymcintyre/RoboticsApplicationDevelopment/tree/master/Task%209%2010%2011%20-%20Project/2.%20Automatic%20Changing%20of%20Lanes/scripts/

To run the feature, the following commands will be required:

```
$     roscore
$     rosrun rosserial_python serial_node.py /dev/ttyACM0
$     rosrun project zumo_move_forward.py
$     rosrun project zumo_line_detect.py
$     rosrun project zumo_lane_change.py
$     rosrun  topic_tools  mux  /zumo/cmd_vel  /zumo/1/cmd_vel
/zumo/2/cmd_vel mux:=mux_cmdvel
```

This feature works by detecting the object ahead using the forward proximity sensors, at distance, then moving into position, performing a right turn followed by a left to move into the next lane and moving forward again until the left proximity sensor no longer detects an object and returning to the original lane. The sensing node (LaneChange.ino) passes to the sensor topics to the perception and planning node (zumo_lane_change.py) that are multiplexed into the control node alongside the other nodes.

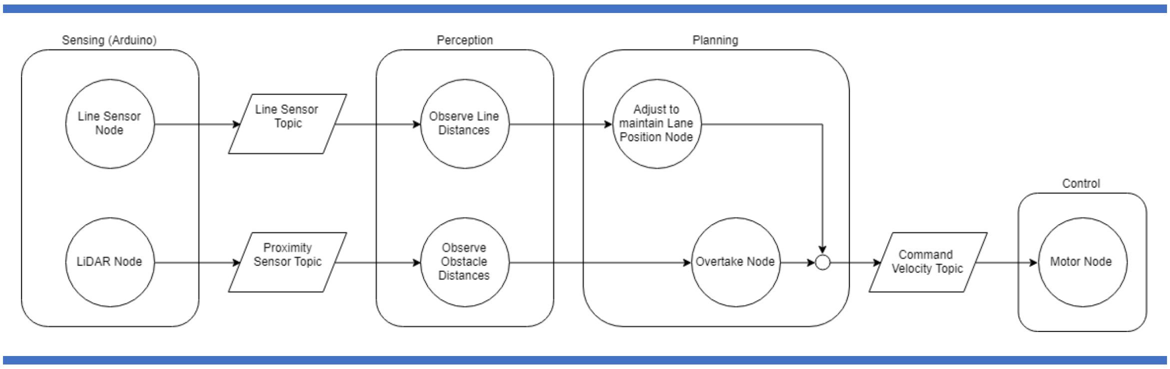


Figure 6: Lane Change ROS Diagram

Conflict Resolution

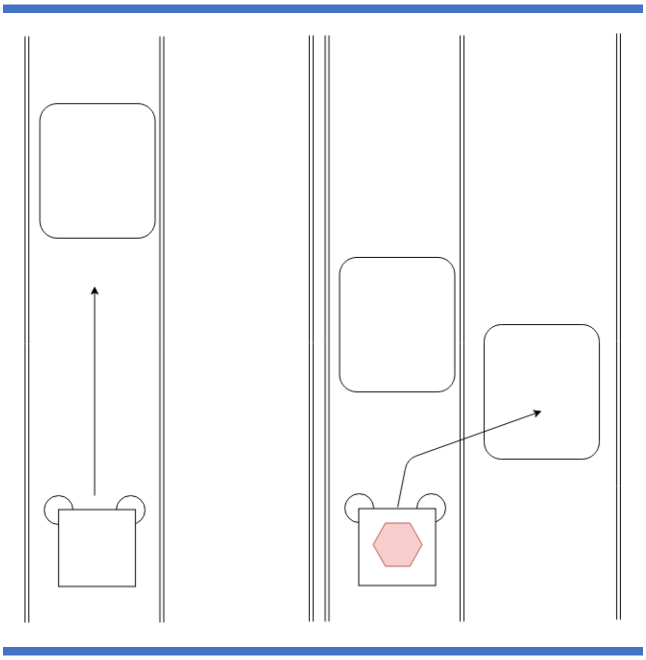


Figure 7: Conflict Resolution

Conflict Resolution is a very psychological topic, the making of complex decisions is not an ability we have yet to give our robotic creations because it is to difficult to program morals and complex decision making that could involve lives (Saffarian et al., 2012). Under scrutiny, many decisions can be defined as bad or good post event. Full information is not always available when a decision is required, triage choice must be made with only the information available at the time.

The Arduino code for the Zumo 32U4 can be found here:

https://github.com/gregorymcintyre/RoboticsApplicationDevelopment/tree/master/Task%209%2010%2011%20-%20Project/2.%20Automatic%20Changing%20of%20Lanes/LaneChange/

(As per the last feature, no changes are required)

The required Scripts can be found here:

https://github.com/gregorymcintyre/RoboticsApplicationDevelopment/tree/master/Task%209%2010%2011%20-%20Project/3.%20Conflict%20Resoultion/scripts

To run the feature, the following commands will be required:

```
$     roscore
$     rosrun rosserial_python serial_node.py /dev/ttyACM0
$     rosrun project zumo_move_forward.py
$     rosrun project zumo_line_detect.py
$     rosrun project zumo_lane_change.py
$     rosrun project zumo_conflict_resolution.py
$     rosrun  topic_tools  mux  /zumo/cmd_vel  /zumo/1/cmd_vel
/zumo/2/cmd_vel mux:=mux_cmdvel
```

For this feature, I have very simply implemented an emergency stop. If an object is detected to be within very close proximity to the robot, then it will immediately halt all movement and wait for further instruction. Many unacademic articles reported on the Uber Automated vehicle fatality in Arizona, United states during March of 2018, the company did not release official documentation but in many customer correspondences, they said that 'swerving' was too unpredictable and could be caused by false positives adding to risk. Thus, this module learns from the unfortunate loss of human live and will not implement a swerve.

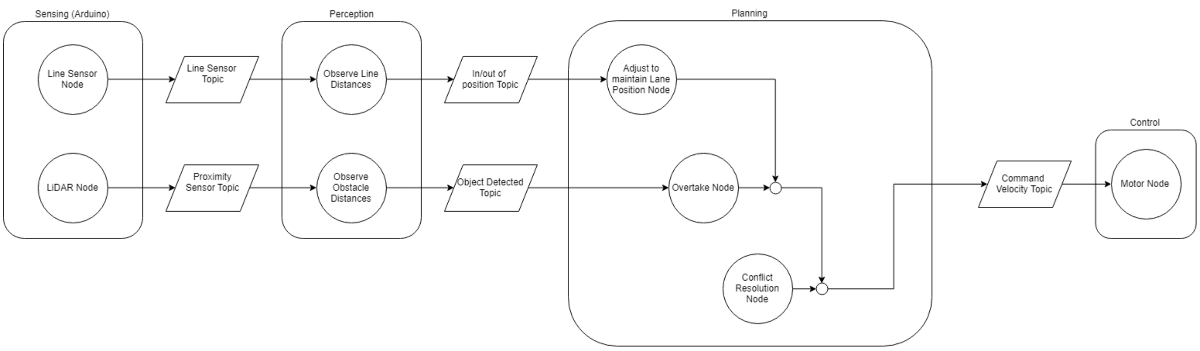


Figure 8: Conflict Resolution ROS Diagram

Conclusion

In conclusion, this report presents detailed information on a great deal of content. A breakdown of the problem that automated vehicles must overcome and why it is difficult to solve given current technology and knowledge, this report found that the SAE has defined six level of vehicle automation, each presenting various challenges, level 0-4 are at a point where they can be solved with current technology but knowledge is lacking for the latter parts from 5-6 the problems become more technical and development of more efficient methods of data sensing, perceiving and planning is required. This report analyses the DARPA automated vehicle analysis architecture to show links between requirements and the risks of not completing them, showing many risks flow down the chain and can have a detrimental effects on other parts of the process.

The report contains an explanation of the Robotic Operating System (ROS) architecture and how It can be applied to these problems explaining the node and topic functional architecture can be utilised to great effect to create and add features with minimal disruption to the existing system.

The report presents three advanced automated driving feature discussions; Navigation using Line Sense, Automated Lane Changes and Conflict Management. It provides a cohesive and through explanation on how ROS could be used to implement a solution, including links to solutions, operation diagrams and ROS diagrams.

The task of creating a level 5 full automated vehicle is a daunting task for a single developer, but as all things in engineering, if we are to approach in a modular and systematic way attacking small parts at a time, it is a problem that is not too far of being solved. Maybe sooner that we expect we could see the driverless taxis of ‘The Living Machine’ or the positronic brain cars of ‘Sally’.

References

BARABÁS, I., TODORUŢ, A., CORDOŞ, N. & MOLEA, A. Current challenges in autonomous driving. IOP Conference Series: Materials Science and Engineering, 2017. IOP Publishing, 012096.

CAMPBELL, M., EGERSTEDT, M., HOW, J. P. & MURRAY, R. M. 2010. Autonomous driving in urban environments: approaches, lessons and challenges. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences,* 368**,** 4649-4672.

CASNER, S. M., HUTCHINS, E. L. & NORMAN, D. 2016. The challenges of partially automated driving.

COPPOLA, R. & MORISIO, M. 2016. Connected car: technologies, issues, future trends. *ACM Computing Surveys (CSUR),* 49**,** 46.

MÖSSINGER, J. 2010. Software in automotive systems. *IEEE software,* 27**,** 92-94.

NORTON, A., OBER, W., BARANIECKI, L., MCCANN, E., SCHOLTZ, J., SHANE, D., SKINNER, A., WATSON, R. & YANCO, H. 2017. Analysis of human–robot interaction at the DARPA Robotics Challenge Finals. *The International Journal of Robotics Research,* 36**,** 483-513.

POLOLU ROBOTICS & ELECTRONICS. n.d. *3.5. Front sensor array (line and proximity sensors)* [Online]. Available: https://www.pololu.com/docs/0J63/3.5 [Accessed 27th of May, 2019].

SAE. n. d. *Levels of Driving Automation* [Online]. Available: http://www.sae.org/misc/pdfs/automated_driving.pdf/ [Accessed 21st of May, 2019].

SAE ON-ROAD AUTOMATED VEHICLE STANDARDS COMMITTEE 2014. Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems. *SAE International*.

SAFFARIAN, M., DE WINTER, J. C. & HAPPEE, R. Automated driving: human-factors issues and design solutions. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 2012. Sage Publications Sage CA: Los Angeles, CA, 2296-2300.