Lane Direction Assist

(An ADAS concept system)

Eric Butler

20094078

Brendan Jackman

Supervisor

I certify that this assignment is all my own work and contains no plagiarism. By submitting this assignment, I agree to the following terms:

Any text, diagrams or other material copied from other sources (including, but not limited to, books, journals, and the internet) have been clearly acknowledged and referenced as such in the text by the use of ‘quotation marks’ (or indented italics for longer quotations) followed by the author’s name and date [e.g. (Byrne, 2008)] either in the text or in a footnote/endnote. These details are then confirmed by a fuller reference in the bibliography.

I have read the sections on referencing and plagiarism in the handbook or in the SETU Plagiarism policy and I understand that only assignments which are free of plagiarism will be awarded marks. I further understand that SETU has a plagiarism policy which can lead to the suspension or permanent expulsion of students in serious cases.

Signed: Eric Butler

Dated: 31/12/2023

Introduction

My project is to investigate and create a solution to the following questions:

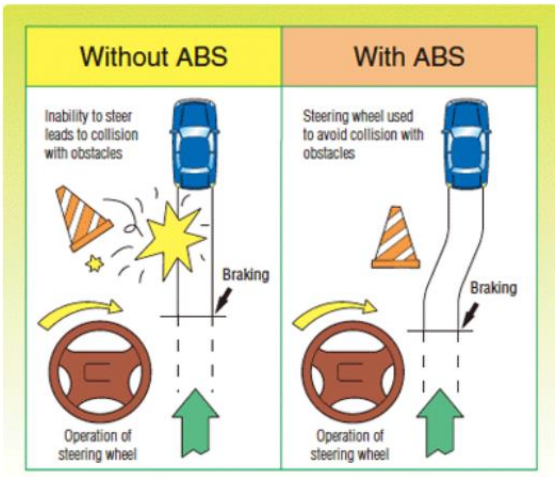
“How can we make roundabouts and junctions safer?”

“How can we incorporate ADAS systems combining with the already existing sensors and technology in order to deliver more awareness and safer control to vehicles and situations”.

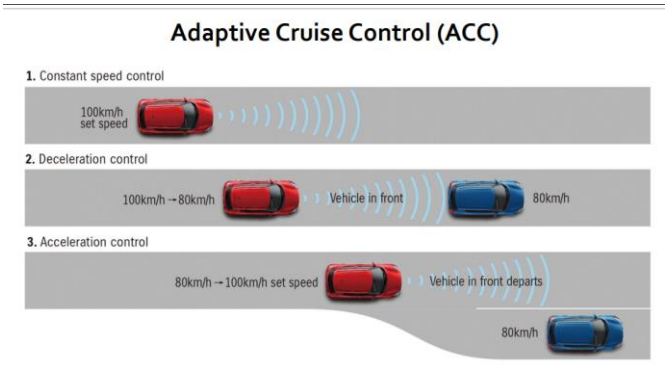
I want to answer these questions by demonstrating my knowledge of different tools and skills I’ve learned through both my automotive modules and placement.

An ADAS system stands for “Advanced Driver Assistance Systems” which are developed to ensure as many situations are catered for in terms of safety while driving. ADAS systems work to Automate tasks such as breaking with Emergency brake assist faster than a human to allow vehicles to adapt to situations on the fly. ADAS systems can kick in, in many different parts of a vehicle, including but not limited to:

- ABS (Anti-lock Braking System): This system ensures that when a user presses the brake, the brakes will not lock and cause the vehicle to slide straight or turn into a skid. Without ABS, applying steering while braking causes this lock. Here is a diagram of how ABS works.



- ACC (Adaptive Cruise Control): ACC is a system that ensures a safe speed is always maintained, if a speed limit is 100 KPH, the system will ensure that if the vehicle is close to 100 KPH, it will stay at or just below this speed. Should a vehicle be present to the car, the system will slow a car down to ensure a safe level of distance and braking time to avoid collision. See image below for diagram on Adaptive Cruise Control.



- LKA (Lane Keep Assist): Lane Keep Assist ensures that a vehicle stays in the correct lane at all times. Should a vehicle start to drift over the line onto the opposite lane, the system will notice through sensors that it has crossed over the line and will start to automatically correct the steering so the car stays in the correct lane. Below is an image of an example of lane keep assist

To develop this concept could include using the same or similar software for things like simulation testing (CANoe, MatLab), ADAS System Manipulation (Manipulating the existing sensors), Car2X technologies to let cars connect to each other in order to demonstrate / simulate cars performing different tasks based on the given situation (turning the incorrect way on a roundabout or junction)

Why I've chosen this project.

My initial project idea was to make the perfect world traffic light system, where I would program a simulation where all cars at a traffic light stop would all move in perfect direction, at perfect speed, distance etc. without ever meeting one another. This was going to be to show if we had fully autonomous vehicles, what a traffic system could look and likely lead to a decrease in cars stalling, crashing, going too fast / slow, people jumping lights etc.

Although after talks with my supervisor, we came to an agreement of not only would this turn to be more of a maths simulation than a fully automotive simulation but it was also a project that would've

been way out of my knowledge range for a student. This led to the result of me ditching this idea and me and my supervisor checking what project may be more suitable.

It was then my supervisor approached me with this project idea of looking at the microscale of my previous idea and adapting it to something like a roundabout or a junction. This project would show ways of increasing the safety of these areas of the road where a lot of collisions can occur. This is where with some communication back and forth that we came to the idea of this project where we can manipulate or trigger certain ADAS systems in a vehicle, for them to kick in when the vehicle is to take the wrong turn onto the wrong side of the road, such as the right side in Ireland or the wrong way around the roundabout. This system would alert drivers that the wrong path has been taken and to turn onto the correct side. Other systems would also kick into place such as lights or sounds in order to signal these further if the correctness isn't taken care of in time.

This idea was agreed upon as it would take aspects of my previous idea as well as shift it more towards the automotive side of production in order to demonstrate my skills in both programming and my automation system modules.

Functional Summary

What I plan to do to carry out this investigation and development on includes the following although this may be subject to change:

- Research what software and tools that I may use to develop this project. This could include:
  - Here GPS mapping (JavaScript map layout) or Apollo open-source virtual environment (this seems to currently be down right now when trying to clone an example repo – 9/10/23).
  - This GPS mapping software should allow the user to see the following: Green path for correctness of how to turn onto a given roundabout or junction, Red path to show where the user cannot turn onto as it is the incorrect way of proceeding safely and lastly an Orange or Yellow path to show the user where to be aware of for precautions before proceeding onto the correct line.
  - If a user does proceed to go onto a path that is deemed red or incorrect, first concept will be to signal the dash to notify the user that they have turned the wrong way. This may follow with a buzzer or some sound effect like a buzzer or beep sequence (may use Arduino for trying proof of concept)
  - Need to do some research into Ziggbee IOT device to look into checking how to simulate Car2x technology by simulating 2 cars talking to each other. This may make it possible to perform a real life showcase of this project in project 2
  - Car2X experiment done with Brendan in order to develop a prototype and proof of concept. Showing how routes and communication between 2 cars works
  - CAPL code and Syntax, this can be looked into when looking at the experiment.
  - More information on Car2X VN4610 in how you can set it up in other ways.
- Develop tools and requirements include:
  - Doing specification document and calendar to track progress and to break down the project into easy to manage pieces.
  - Hardware tools such as the VN4610 again to see how it works in the scenario I want it in.

- Software tools such as Car2x simulation, this can be done alongside my assignment.
- Here mapping to maybe have GUI features within the project
- CAPL code to ensure simulation runs correctly

Assumptions

- VN4610 should relatively be plug and play with CANoe Car2x
- Proof of concept may not include and hardware demos.
- Light concepts may only work with HERE Developer.
- Here developer may be difficult to connect to CANoe.
- Procedure of simulation should run smoothly without issue, if configured correctly.
- Traffic light system possibly may not be implemented in this iteration of the project as it's not essential and is more for complexity.

Design Objectives

- May be worth it more too just go with Car2x routing if HERE Developer doesn't work out. Otherwise I could try find similar software in order to fulfil the needs of the project.
- Software over hardware: It may be more time efficient to prioritize software testing and concepts over hardware. This will allow me to create the best possible project and not get caught up in trying to do both.
- Don't focus on nonessential features: Prioritize the prototype with essential features first, such as the logic behind the cars meeting the RSUs instead of looking heavily into something like a traffic light system that is more for complexity rather than to further show proof of concept

Functional Requirements

System Context

Interface of the map showing the cars moving and colour coded directions on a junction / roundabout to show correct way of travel. The interface will also show signals if the car turns the incorrect way on the given route, possible show a message to show it has turned the correct way.

Requirements List

Some CAPL functions may include:

- Function for position of car on map
- Functions for transmitting the signals between the RSUs and the Car nodes such as car speed? , wheel direction (if this is possible), indicators (if possible)
- RSU to maybe simulate traffic light, so it can pick up signals from the car. Additional functions could have the light indicators (Red, Green, Blue), Timer for when the light will change from Red to Yellow to Green etc. Possibly setup an interface for this.
- Multiple Routes:
  - ➔ Route with correct path to signal the driver that they have turned the right way through a signal
  - ➔ Route with incorrect path to signal driver they have turned the wrong way, possibly multiple signals shown to the driver depending on how long they are on the incorrect path for.
  - ➔ Route to show a driver going around a round about the wrong way.
  - ➔ Route to show driving down a one way street

- Network parameters to allow CAN, DENM, Stacks etc.
- Possibly binding some of the functions to 'on key' to allow for user interaction. (Possibly not worth it for real life demo).

Interface Specifications

For my project 1, I have will have a CANoe simulation where it will have interfaces of a roadmap similar to google maps where a car will drive along a path and turn onto a wrong road and proceed to drive down this road. Interfaces will be setup to show possibly blinking lights as well as some code which will prompt messages in the console stating that the car has turned onto the wrong direction.

As well as this I hope to also have some other interfaces like possibly a traffic light system as well as the colour co-ordinated road as mentioned above, though these may not make it into this iteration of the project. This is due to the traffic light not being an essential part of the project and so it is a complexity element of the project. Even though I may not fully understand how to implement this, I may possibly simulate as if the car was to stop at a traffic light on the map where it is in person.

The network protocols put in place will be the network messages transmitted between the cars and the car. These network messages can contain things such as the vehicle speed, location, distance to car in front / behind. Distance to RSU (Roadside Unit) to transmit certain actions.

The RSUs will be used for things such as car detection, possibly a traffic light simulation, as well as if I was to implement counters for cars passing, it could be implemented through an RSU. In terms of this project, at least from project 1, there may not be much if any RSUs in this iteration.

Non-functional Requirements

Looking at the non-functional requirements, there will be some constraints / requirements for this project to be run. Firstly, the user must have a licence to the Vector CANoe software, this being a full licence which I was given supplied by SETU as part of my course or a demo licence which can be downloaded from the Vector website. This Software is only available on Windows and as of right now it isn't supported for MACOS.
As well as this, a package with the licence may also be required, this being the Car2x package. This package as well was supplied to me by SETU for the purpose of both my modules and my final year project. This is the core fundamentals behind my project idea and so is crucial to the operation of the demo.

Some more constraints to the project may be the restrictions to the software, as experienced before there are some bugs with the software most notably that you cannot have more than 2 scenarios at once as this can cause the application to crash more then not and so I would like to minimalize this issue as much as possible. This issue can also occur if you have too many vehicles / RSUs in a scenario at once as experienced with an assignment that also used this software.

Quality assurance provisions

Software test procedures

For project 1, my main testing procedure will be more so the simulator in the fact of all my lights / car movements work as intended. With CAPL it is difficult to write standalone tests which is what my project 1 will be based upon so for this reason I will try implement things such as, error handling, correct console logging or system variables that throw actions when events occur.

Although my testing may be limited due to the limitations of the software, This is a key point to note and to develop upon come Project 2 and once I've had some talk with my supervisor on this issue

Software validation procedures

Similar to the software test procedures, the "test" procedures will work similarly as validation procedures again, by testing my simulation functionality by setting up some log commands or system variable actions that trigger upon event occurring. This could be something as simple as a light starting to blink as the car gets to a certain point. Similarly, mentioned in the test procedures section, I will take this as a key point into project 2 and touch upon this more, this could be something along the lines of testing this simulation with an actual car in a test environment and ensuring that all functionality works correctly.

Methodology Overview

The methodology approach I used for my project was the V-Cycle methodology, this entails, outlining the requirements of the project and developing each section of the project and testing along the way. During my project, some of my activities performed were:

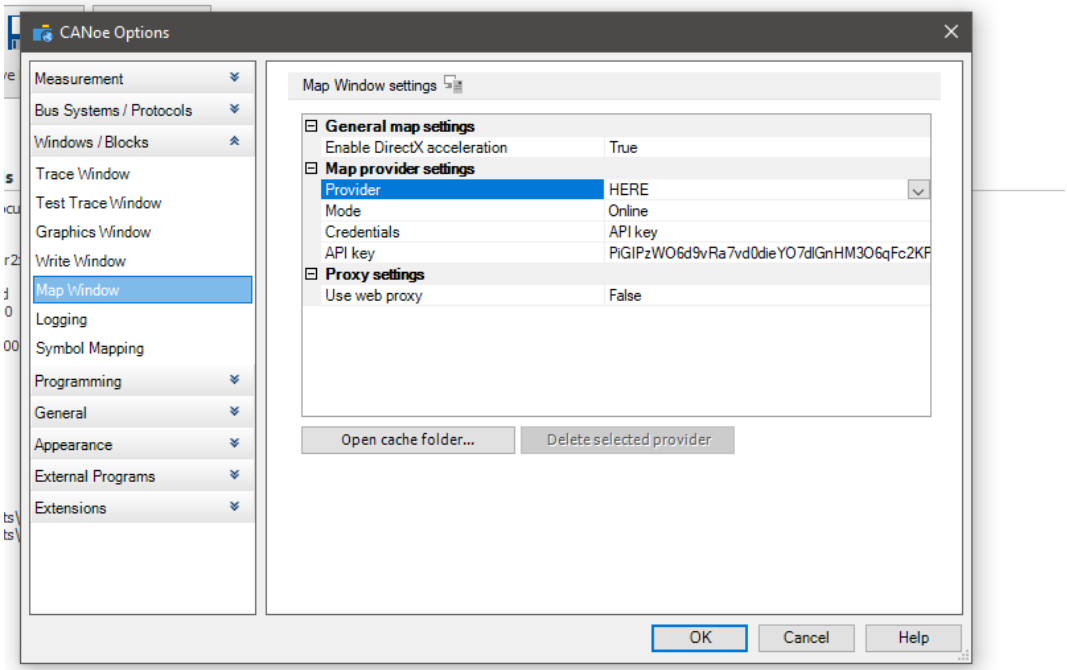
- Researching ADAS systems to learn about what my project would include: Firstly, I was introduced to ADAS systems through a module within my Automotive Stream and this is where I start to think about my project being based around it. I would have to partake in my classes related to this topic as well as research some more information on the side of how exactly these systems work and which systems I'm going to focus on when thinking about my idea and design. This included looking at systems such as EBS (Emergency Brake System, Collision Detection Systems, and Anti-Lock Braking Systems as well as Traction Control systems.

- Researching Documentation for CANoe Car2x: This is where I started getting experience with CANoe Car2x by either following along through notes during my module, completing assignments based on this software as well as reading up on documentation / watching tutorials provided by Vector to gain more knowledge about the software. I would then go on to develop some simulations using different features / techniques I had learned through assignments, documentation, and tutorials.

- Researching HERE Developer mapping: I performed a similar approach to the CANoe software but for HERE Developer, although none of my projects didn't require me to use this software, I did download some examples from the website and experimented with them. After some extensive research of watching some tutorial videos as well as looking at documentation as well as trying to implement HERE into CANoe, I came to a conclusion that it was taking up too much of my time for a feature that wasn't really essential and agreed with my supervisor that I should drop this idea for now.

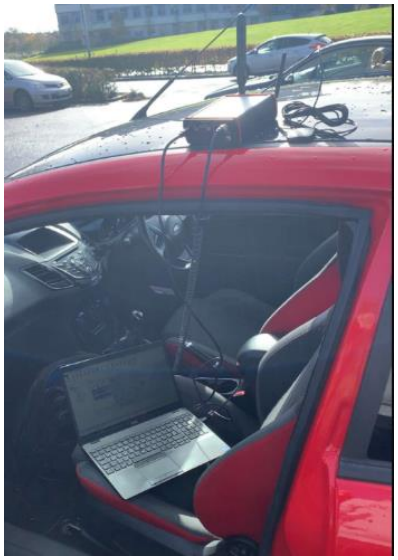
  Here are some screenshots of some examples I looked at. This is an example of a 3D map render of Waterford done through JavaScript.

In this screenshot, on the right we can see a 3D render of a map in which there is a tilted angle, and on the left some JavaScript and HTML code that prints out this map. To change the location, you simply plug in your desired co-ordinates and it will show a 3D render of the given space. The reason for choosing this to research was to possibly experiment with the idea of colour co-ordinated roads that will show the user the correct approach to the given road. It would also allow for a more pleasant viewing of the system to someone who may not be familiar with CANoe. The HERE system has direct integration with CANoe although I could not get this to work and came to a conclusion with my supervisor that maybe I should not pursue with this right now.



Window showing the HERE developer provider integration using API key

- Experimenting with the Vector VN4610: The VN4610 is a piece of hardware that can connect to a computer through USB as well as into a car through a lighter socket. This hardware piece can act as a network node and capture different pieces of data as a car functions. This data can be GPS, Route taken, speed, gear changes etc. During a practical class we experimented with this, and I got some hands-on experience where we got 1 car to drive around the block of the campus and another car staying stationary in the car park, both cars having these VN4610's connected to them. A log file was created that showed all the details mentioned above. During my second iteration of my project, I plan to use this hardware more.



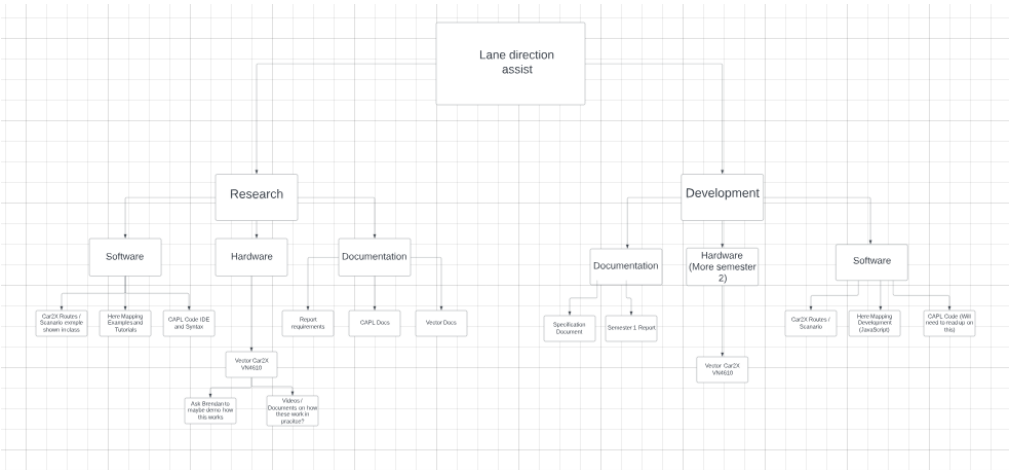
Car setup with the VN4610 to capture Car2x measurements

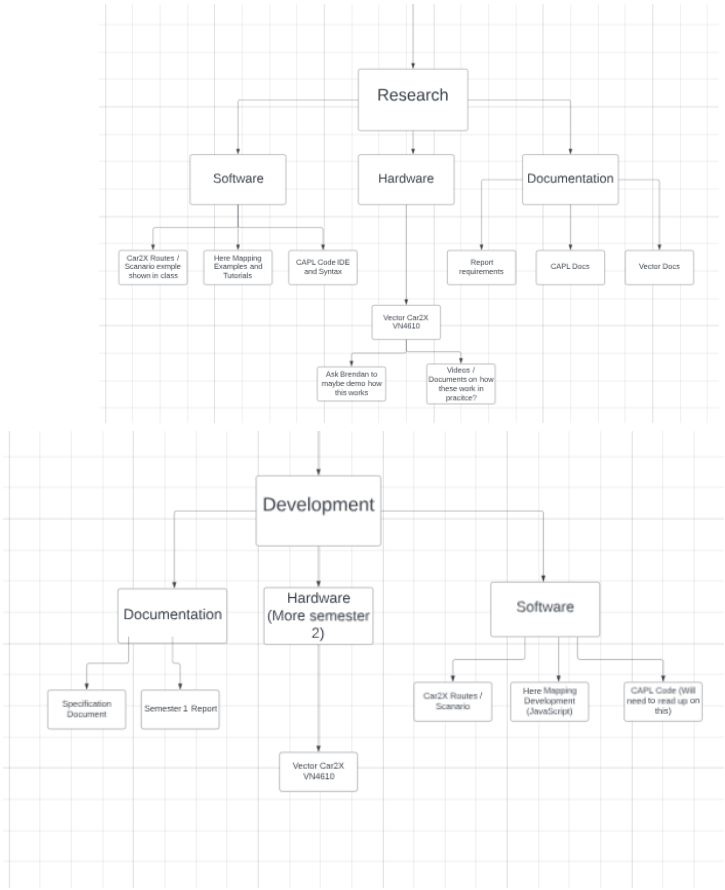
- Experimenting with the associated language (CAPL): Through assignment work throughout a module, I would get familiar with the associated language related to CANoe which is CAPL. This a language similar to C that is designed to integrate with the different objects within CANoe such as Network nodes, System Variables, CANN / DENM Messages. Using my small experience with C, I was able to get familiar with these concepts and use them to develop the prototype related to this project. More details explained below in the "CANoe Simulation Description".

Project Plan

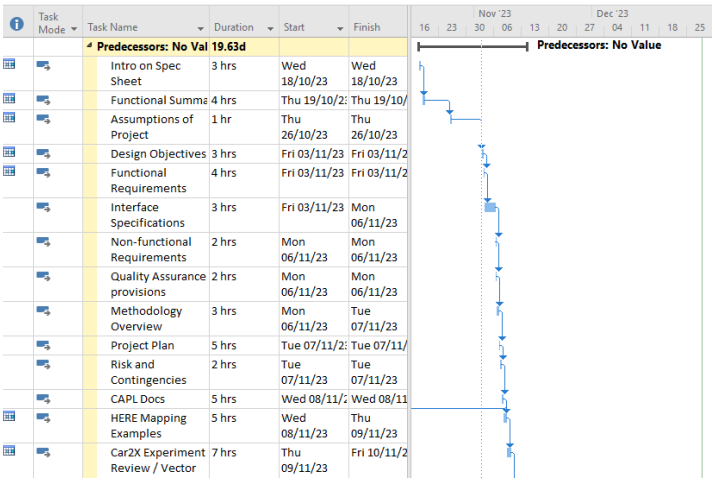
Here I will explain my project plan, this will be similar to the methodology overview but going into some more details on specific tasks. I will explain my breakdown structure as well as provide some screen shots of the Gantt charts created.

1) Firstly I started by thinking about what my project would include in terms of research and development. This is where I started to think about the individual tasks to the micro level and developing an ER Diagram to visualise my plan process. Here are some examples of the ER Diagram.





2) Secondly, I started to develop a project plan through Microsoft project where I could put in my schedule, my tasks that would have to be completed as well as a rough ball park on how long it would take me to complete each task. This then scheduled all the tasks to be performed as well as a Gantt chat you can see below. This made things easier to follow and to keep on track with what was needed to be done. Although there are start and finish dates, these would very much fluctuate as there was a lot of extra time taken up for assignments / lab work from modules etc.



Task Mode	Task Name	Duration	Start	Finish
	Predecessors: No Val	**19.63d**		
	Intro on Spec Sheet	3 hrs	Wed 18/10/23	Wed 18/10/23
	Functional Summa	4 hrs	Thu 19/10/2:	Thu 19/10/
	Assumptions of Project	1 hr	Thu 26/10/23	Thu 26/10/23
	Design Objectives	3 hrs	Fri 03/11/23	Fri 03/11/2
	Functional Requirements	4 hrs	Fri 03/11/23	Fri 03/11/2
	Interface Specifications	3 hrs	Fri 03/11/23	Mon 06/11/23
	Non-functional Requirements	2 hrs	Mon 06/11/23	Mon 06/11/23
	Quality Assurance provisions	2 hrs	Mon 06/11/23	Mon 06/11/23
	Methodology Overview	3 hrs	Mon 06/11/23	Tue 07/11/23
	Project Plan	5 hrs	Tue 07/11/2:	Tue 07/11/
	Risk and Contingencies	2 hrs	Tue 07/11/23	Tue 07/11/23
	CAPL Docs	5 hrs	Wed 08/11/2	Wed 08/11
	HERE Mapping Examples	5 hrs	Wed 08/11/23	Thu 09/11/23
	Car2X Experiment Review / Vector	7 hrs	Thu 09/11/23	Fri 10/11/2

	Car2x Route Scanario	10 hrs	Fri 10/11/23	Mon 13/11/23
	HERE Mapping 3D Render of Waterford	10 hrs	Mon 13/11/23	Tue 14/11/23
	Predecessors: 13	***0.63d***		
	CAPL Code	5 hrs	Wed 18/10/2	Thu 19/10/

3) After I had both my ER diagram and my Project charts laid out, It was time to start writing up an introduction to the document to ensure I was happy with the idea and how it sounded by writing it down. As well as this I started into just some small research looking at the lecture slides for my module, watching some explanation videos on what ADAS is as well as just gathering any information that may be deemed necessary to carry out this project.
4) I started to fill out my functional summery to get an idea of the different functions that my proof of concept would require. Some of these concepts and ideas were quickly thrown out with some for e.g. the HERE mapping, would go onto further research until either being part of the project or scraped at least for this iteration.

5) After I had gathered the functions that would be required for the project, I started to write down assumptions of how the project would work, these could be very obvious assumptions of just the software needed to having uncertain assumptions such as the complexity of integrating the HERE developer environment into CANoe.

6) After creating my assumptions, I started to look into examples of the HERE mapping, here I looked at examples that allowed for a user to import a co-ordinates into a JavaScript environment and it would create a 3D render of the given location. More information on this in the methodology overview. As well as this I was also writing my Objective details on which objectives I'd tackle first and in what order.

7) As well as continuing on with my research on HERE, I also started to write up my functional requirements which would consist of a deeper dive into the functionality of each task that would eventually make up the project, this can be something like thinking about what the individual cars would do to make up the full simulation. This allowed me to break up the workload required and stay focused on getting the project together.

8) Following from this came my interface specifications. This is where I would determine What to expect from the visualisation of the project, This being things such as the cars driving down the route on the map, a blinking LED as well as some code printing some statements. There was some interfaces that were considered but not built upon in this iteration.

9) Once I started to finalise how my project will function, it was important to diagnose the non-functional requirements of the project. These are things that will be required to run the project as well as what the project relies on. As mentioned in this section, things such as the license for CANoe, the package for the Car2x properties and objects would fall in line with this sort of criteria.

These are things the project requires but may not be a general function within the software. As well as this could be an account to HERE developer, should I have implemented this feature.

10) With all this in place, on the side in my automotive module I had started to develop different simulations for assignments within this software. I had discussed with my supervisor about gaining experience in the software before starting my prototype. After some dissuasions I would start experimenting with different scenarios on the side as I was doing my assignments. This meant I was learning as I'd go as well as building up my knowledge on the software which would be useful to the simulation.

11) Lastly, I continued my research upon both Vector Software / Hardware as well as discussing ways to possibly bring this project to a real car scenario. Concluding to the fact that this will be done in a controlled test environment and done using something like cones to setup a path that a computer might deem the incorrect path. This will hopefully be able to set off either a signal on the laptop or should the car have a signal built in I can connect it too, it can set off that. As well as this I continued to document my report and follow my Gantt charts as I go.

Risks and Contingencies

Thinking of the risks and contingencies of this project, in terms of project 1 where we are merely talking about a simulation, the only real risks we can talk about are things such as WIFI failure, pc crashing, software not loading / crashing as well as general issues that can arise such as battery failures etc.
The risks that can arise for this project if it was implemented into a real world car can be things such as a car not alerting a driver of the wrong direction in time which could lead them into oncoming traffic. Although the vehicle in my given situation would be going at a low speed, there are still risks to damage the vehicles as well as the people inside them. Similar to this there's other risks of the correct ADAS system not kicking into place in time such as the breaking system should these systems be chosen. Lastly, the last risk I can possibly see being an issue is the VN4610 system not working or breaking upon testing. This being an integral part of my project and testing it's crucial that this doesn't get damaged and works correctly.

To cater for these possible risks, starting with the simulation I will put measures in place such as, having one of the computers in the room ready, should my laptop not function correctly or an issue with the software doesn't load. Another parameter I will need to ensure is that I have another WIFI connection ready should the internet in the building go down during my presentation.

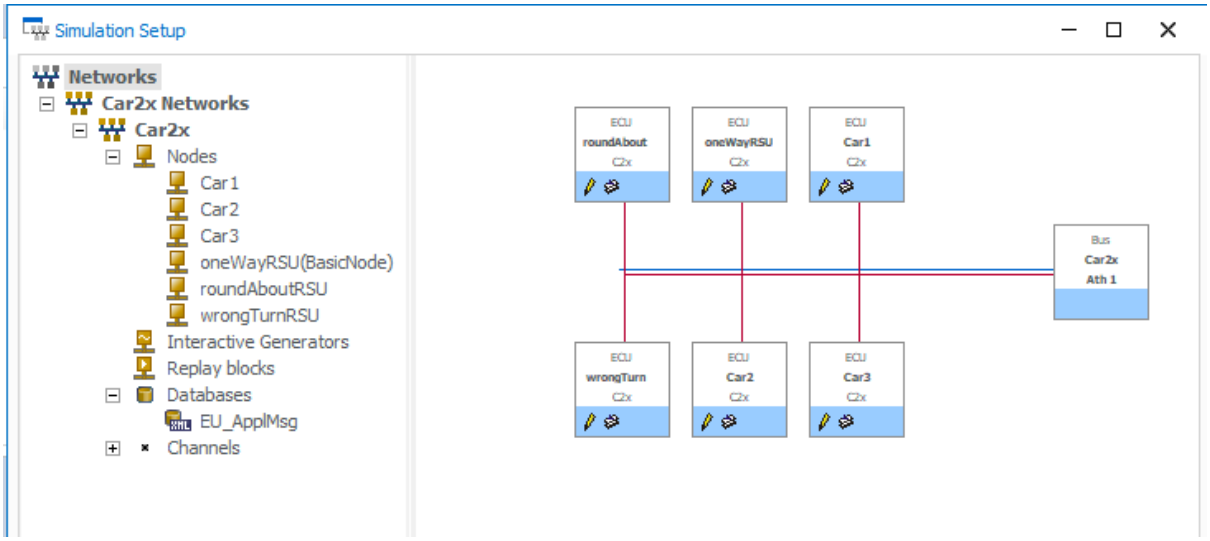
To cater for the real world scenario risks, I will perform any tests needed in a setup controlled environment where the only car present will be the car in which the system is being tested on. Also to carter for the VN4610 staying intact, I will ensure that it is secured to the top of the car and all power sources are plugged into the mains correctly.

CANoe Simulation Description

Vector CANoe is an industry standard software that allows users to create scenarios for vehicles with their Car2x package. The software is used mainly by automotive manufactures' for simulation, development, testing, analysis, and diagnostics for ECUs (Electronic Control Units) within vehicles.

The software caters for many different scenarios that can happen on the road such as heavy traffic, cars breaking down, construction, intersections, etc. which can allow for OEMs (Original Equipment Manufacturer) to find as many bugs or flaws within their systems as possible before the vehicle is shipped to production. This software also has features for simulating dashboard signals and prompts directly built in. In this way, you can simulate different triggers that may occur within a modern car's ADAS (Advanced Driver Assistance System).

Simulation Details



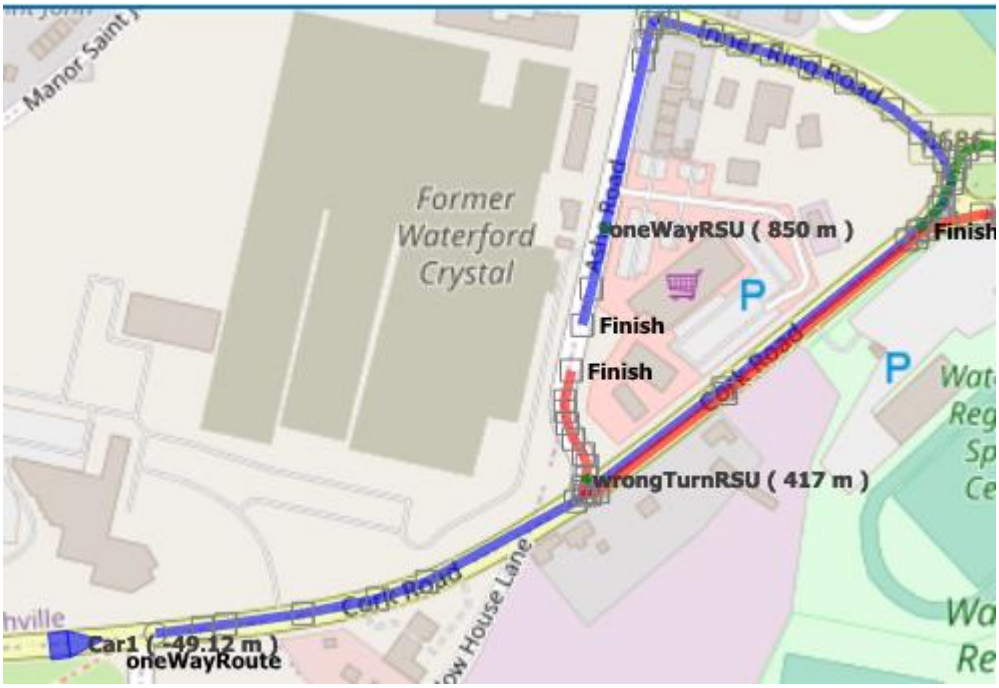
Screenshot showing the Simulation setup which includes Network nodes and names used.

For the purpose of my project, I will be focusing on a specific package that allows me to focus more about ADAS and around the system I want to develop. The package I will be focusing on is the Car2x package to develop a simulation that has multiple cars going different routes where situations will occur such as:

- A car turning onto the one way road from the wrong side: Within the location I will be using, A one way road exists that can only be driven on from the merging lane. Here I will have a car turn onto this road from the oncoming lane and therefore a RSU will pick this up and perform an action such as possibly making a light blink / illuminate.

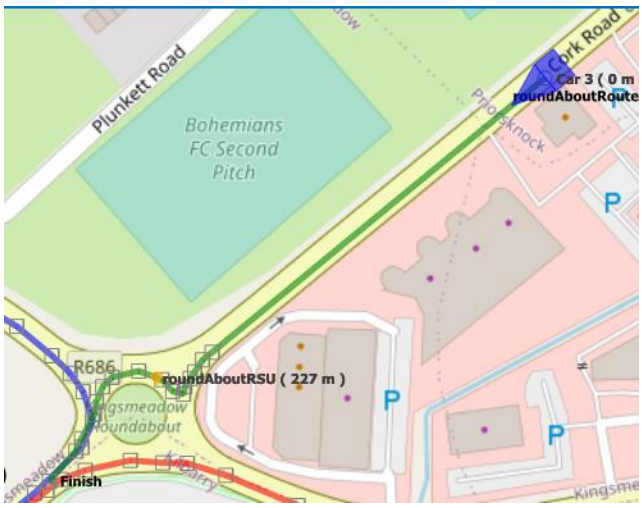
Represented in red is the path that will lead onto the one way turn from the wrong direction.

- A car proceeding down a road onto the one way section: Another route I will develop will be a car driving down a road but will go beyond the point of direction allowed. Here the car will go down a one way section of the road before again setting off a light and then possibly slowing the car down.



Represented in blue is a route where a car will proceed longer then the road would allow and onto the one way street

- A car proceeding around a round about the incorrect way: Here I will configure a car to come up to a roundabout and start driving around it on the incorrect side. A RSU will be setup on the inside of the roundabout that will monitor the direction of the cars going around and if a car is deemed the wrong way, it will send a message to alert the car.



Represented in green is an incorrect path to a roundabout.

- CAPL Code setup to handle detection functions: The project will include some CAPL code that will act as a ruling system to detect the cars given a certain criteria. This code will hold parameters such RSU details, Vehicle details such as type, speed, and the distance to the RSU. This will be to ensure that the correct triggers occur when an RSU detects a car operating incorrectly.

```
        // Calculate the linear distance to the CAM station whose message was received.

        distanceToCam = API_PosCalcDistance( vehicleCamLat,  vehicleCamLong, camLat, camLong);

//        write(" *** CAM RECEIVED *** Station Type = %d Long = %f  lat = %f distance away = %f \n", camStationType, camLo
//        write(" *** station Id %d speed = %d heading = %d \n", camStationId, camSpeed, camHeading );

 //   Signal that a stationary RSU (camStationType )has detected the cam
//      write("Received CAM - Station Type: %d, Speed: %d, Distance: %f", camStationType, camSpeed, distanceToCam);
    if (distanceToCam <= 50 && camSpeed == 0 && camStationType == 0)
    //&& camSpeed == 0 && camStationType == 13
        {
          write("test");
          @sysvar::distanceToCam::distanceToCam = distanceToCam;


        }
        else {
          write("searching for rsu");
        }
  }
}
```

Although not the final code, this is a sample of what the CAPL code can look like. Above shows us declaring "distaanceToCam" (Cam = CAMM Message = RSU) and we intake the vehicles position as well as the RSU position. We then write a rule where if the car is less than or equal to 50 meters to the RSU, the Speed of the RSU is 0 and the stationType is 0 (which is the number to different objects within CANoe) then we print out a test message, and we print

a system variable message to an object which in this case could be a blinking light or warning sign.

Otherwise we constantly print that the car searching for the RSU.

Conclusion

To conclude this report, there are many things I have learned about my idea and proof of concept that I can take into my 2nd iteration and project. Some of these things are fundamental features to the project and some are just some overall personal lessons to take away from the project in terms of what I personally think went correctly and what maybe didn't go to plan upon my initial thoughts and ideas.

Some of my positives to take from this iteration of the project include:

- My proof of concept and idea is very much possible and discussions with both lecturers as well as industry staff through job interviews have more than confirmed and backed me on this statement.
- My learning outcomes and overall the new technologies I've learned from researching and the small pieces of development have opened up new skills and opportunities to what I may possibly be capable of in the future.

- Knowing when an idea I may have thought about integrating isn't going correctly and the best approach is to move onto something else and either ditch it entirely or return when I have better understanding.

- Project planning and how to take an idea from just a thought process in my head or a lecturer asking me a question, into a full solution and project roadmap that will hopefully take into part of development.

Here I would also like to discuss some things that maybe didn't go to plan or what I felt like I could've done better.

- Time management became a massive flaw for me due to the overwhelming amount of work this semester, Although I tried to dedicate as much time as possible to this project, I do feel like I maybe could have put even more time in. Although it is important to note that this iteration is more focused on the research and proof of concept rather than the development as of right now.

- I also feel like my prototype could have been more complicated. Maybe it was due to the fact of me not taking to the software as quick as I thought I would, or not understanding enough the things I wanted to implement, I definitely feel my prototype was lacking and didn't fully reflect my ability.

- The amount of new software and technologies I researched was also a step down from my initial idea. I personally feel I could've and should've reached out more in terms of looking at different things to include such as, multiple programming languages, more environment, technologies etc.

For project 2, I definitely I can take all these learning outcomes both positive and negative and improve upon them to ensure my project is the best version it can be and done to the best of my own ability.

References

Vector Group | Vector (no date) *Vector CANoe Car2x*. Available at: https://cdn.vector.com/cms/content/products/canoe/_car2x/Docs/CANoe_Car2x_Product_Information_EN.pdf (Accessed: 2 October 2023).

Jackman, B. (no date) *Log in to the site: SETU Moodle, ADAS System Introduction - ADAS Systems*. Available at: https://moodle.wit.ie/course/view.php?id=199953 (Accessed: 11 September 2023).

Mobileye (no date) *Responsibility-Sensitive Safety (RSS) a model for safe autonomous driving, Mobileye ADAS Safety*. Available at: https://www.mobileye.com/technology/responsibility-sensitive-safety/ (Accessed: 4 November 2023).

JSFiddle (no date) *HERE JSFiddle Tilted Map*. Available at: https://jsfiddle.net/gh/get/jquery/2.1.0/heremaps/maps-api-for-javascript-examples/tree/master/tilted-map-bounds (Accessed: 18 September 2023).

Link to ER Diagram for project outline

https://lucid.app/lucidchart/4acaa430-3c8f-4509-90dc-c03c746b75ed/edit?invitationId=inv_155ff20a-2b06-4e57-b7d8-4210253afb2e&page=0_0#