Script.
https://docs.google.com/presentation/d/1u7z7WLBv9GHjkwRHIIL8S9FYReycmCTsxKgla6FO5oU/edit#slide=id.g123a5620f80_0_10

INTRO: includes problem statements (Start / Slide 1)

Hi my name is Will from Group 6, and welcome to our presentation for our Fuzzy Logic Controller for PBL1.

This project was based around exploring the use of fuzzy logic in an automatic vehicle braking system.

Traditional computer logic is generally in the form of binary inputs and binary outputs and requires values to be precise, or absolutely true or absolutely false.

This doesn't cut it with complicated systems that have vague linguistic terms, as braking to avoid an object isn't an on or off situation. The required brake pressure is subject to various conditions, which we have explored during this task.

One of the reasons for having fuzzy logic systems in cars comes from the fact that human reaction time in crash scenarios is slower than computers. Computers can process information hundreds of times faster than a human, and can correctly respond to a crash scenario without driver error.

Second slide: Our approach (Slide 2)

Our approach for this task was to first research all the different Intelligent vehicle control systems involved. We looked into the different systems - such as steering and cruise control - and the factors that influence them. We ultimately decided to focus solely upon implementing a collision avoidance system; through the use of applying brake pressure to avoid colliding with a detected object. Initially, our main considerations were object distance, vehicle speed, road slope and road condition.

We also looked into different technologies we could use to develop our fuzzy logic system, and our model.

We began designing a prototype system using MATLAB to implement the Fuzzy Inference System. Then in Python utilising the scikit-fuzzy library to test scenarios - with a simplified model using only speed and object distance.

After a rough prototype was complete we added road slope and road condition, and began to develop the rule base for our system. We quickly realised that having 4 inputs with 4 membership functions each, drastically increased the number of rules we needed, so we reverted to having just 3 inputs.

We then moved onto tuning the model by incorporating braking distance statistics sourced from the Queensland government to make our model and membership functions more realistic.

Membership Functions (Slide 4)

Membership functions are used by a fuzzy logic system to convert crisp inputs into a fuzzified linguistic variable.

As the overall goal of the system is to avoid a collision - the main inputs for the system are the speed of the vehicle and the distance to the detected object. We also added an additional input of how wet or dry the road is to have the braking system compensate for the effect of liquids on the stopping ability of the vehicle. We originally implemented speed with gaussian membership functions as we believed they would better represent the curve of acceleration and deceleration. However, after experimenting with different types of membership functions, we discovered that using triangular membership functions for both speed and distance produced an output that most accurately reflected the real world data we were designing the model against.

We chose wet and dry for the road condition input and selected both z and s-shaped membership functions to represent them as …

- 5 MFs speed, object distance, braking - why?
    - Why that range for each function?
        - Speed: 0 - 110 km/h

        - Object distance: 0 - 100m
            - The decided range for the object distance and speed both influenced each other. Typically speed limits on roads don't exceed 110 km/hr and travelling at such a speed would take just over 3 seconds to clock 100 metres, adhering to the commonly known 3 second rule.
- 2 MFs road condition

Timing: ~45s

Fuzzy Logic Implementation (Slide 4)

The first step to evaluate a fuzzy logic model is to take the normal "crisp" input values and turn them into fuzzified linguistic variables. This is achieved by passing the input values such as speed or object distance to their relevant membership functions that possess a range of linguistic variables - in the case of speed, 42 km/h is somewhat slow and somewhat medium.

The rules are then applied with the fuzzy values as inputs, resulting in a number of truth values for each rule.

Now that all the rules have been evaluated, there will be a number of fuzzy sets which are then all aggregated into one output fuzzy set.

The final step is to find a final, defuzzified output value from that fuzzy set. For our model, we employed the centroid method. The centre of mass of the area under the line is found and that value is the final crisp output value, in our case the Brake Pressure.

Fuzzy Logic Solution Specifics (Slides 5 - 10)

Slide 6

Here we will go through an example of how our Fuzzy Inference System produces an output for the autonomous vehicle to apply braking pressure.

Can see that the input values are 42km/h for speed and that the detected object distance is 33 metres away. The water detected on the road is at 7.

Looking at the membership function for speed you can see that autonomous vehicles is mostly considered to be moving at a slow speed, and somewhat moving at a medium speed.

The condition of the road is mostly considered to be wet

The detected distance of the approaching object is mostly considered to be close and somewhat considered to be near.

These inputs will satisfy a number of rules that had been set in the fuzzy logic controller: <read rules slide>

Slide 7:

When utilising AND rules in a fuzzy logic system the minimum value will be selected from the combined fuzzy sets.

Slide 8:

<Read values from slide>

Slide 9:

The weight of each rule is determined by the minimum of the fuzzy inputs that make up the rule. Each rule is ‘filled up’ according to its’ rule strength. <Mention how the triangles are filled.>

As you can see it has created a fairly broad distribution of the fuzzy output for applying brake pressure. The brake pressure’s range is from zero - ie no brakes, to 10 which is applying full pressure to braking - or an emergency stop.

Slide 10:
Here we can see the overall solid shape for the output for brake pressure. However the output is still fuzzy and it’s hard to tell what kind of brake pressure should be applied.

Slide 11:
Our fuzzy logic system utilises the centroid method to defuzzify the output and come up with a crisp output value. The centroid method finds the centrepoint of mass for a shape. For this

shape it is shown that 5.04 is the centre and that will be our defuzzified output based on all of the rules that had applied. Therefore given all the inputs at the present point in time the autonomous vehicle will apply roughly 50% pressure to the brakes.

Evaluate Fuzzy Logic Controller (Slide 12 - 13)

Thank you XXXXXX, Hi everyone, my name is Nelson, I am mainly responsible for the python coding, adjusting the rule base and creating the environment simulator. As we all know It takes three inputs; the vehicle's speed relative to the stationary object, the vehicle's distance from the object, and the road condition in terms of dryness and wetness.

Once we have the rules feeded into the interface, we will need to adjust it accordingly so that the brake pressure will be at Maximum according to the data sheet from Queensland Government.

Our next decision is to double the distance according to the sheet and see if the braking pressure appears to be lower, and we keep adjusting the rules until it meets our expectation for different stages.

Throughout the testing and refining of our fuzzy inference system we were able to successfully reproduce an output that mimicked the Queensland Government report on stopping distances in relation to the speed of a vehicle. From the report the conditions of the road were assumed to be on a straight flat dry road, we can finally use this model and put it into the environment simulator. And that is the result if the vehicle at speed 100KM.h and the collision avoidance system is being activated at 100M.

>>> play video

This simulation indicates that our fuzzy model can help to avoid collisions.

[- Minutes]

In this simulation, we only took into consideration the emergency stopping distance of a vehicle once braking was applied and decided by the vehicle controller, as reaction of the fuzzy controller should be immediate once an object is detected. After different speeds were tested using our simulator, the fuzzy inference system was able to apply adequate braking force to safely stop the vehicle and avoid a collision with the detected object.

Introducing the factor of the wetness of the road showed that the system would brake earlier to compensate for the conditions and come to a safe stop. So at higher speeds it would brake earlier to compensate.

Limitations and Constraints (Slide 12)

We constrained the design to just that of the collision avoidance subsystem, independent of the other possible car control systems.
In reality the system that we have designed would be a minute part of a greater whole for autonomously controlling a vehicle.

Our main focus was on the approach speed and distance to a detected obstacle.
We made assumptions that the distance and speed detected are in a straight line - or are at least projected into a straight line - and that the obstacle will always be in the direct path of the vehicle, ignoring objects that suddenly appear in that path, such as vehicles that are executing lane changes.
The braking system also does not maintain speed, it only considers avoiding collisions.

Physical phenomena such as wheel slippage, non-linear friction coefficients on tyres, and weight loading were also not considered.
They are complex and non-trivial effects to model, with maths that doesn't translate very well to fuzzy models and would likely need a lot of extra composition to integrate. They are better off used as modifications to the inputs

[An interesting concept that wasn't considered was ethics and how they would factor into a fuzzy logic system for autonomous vehicles, for example the trolley problem.

We also had to cut down on the number of input variables. We chose to only use 3 of the potential variables - Approach speed, distance, and road condition.
The more input variables, potentially the more rules that are needed to get the system to behave as expected. Of course that entirely depends on the fuzzy set used for each variable as well as the rules used. Through clever use of rules using negated memberships or only using a subsection of the possible variables, the rule base can be cut down and still maintain the functionality.
That does raise complexity and can also make the rules harder to understand, depending on complexity.

as mentioned earlier, this collision avoidance model would likely end up being a part of a larger system. Using composition, we could connect up multiple fuzzy models together, say a cruise control model that takes the current speed and a target speed (potentially as one input) and accelerates or decelerates to match it, then pass that, along with the collision avoidance output over to another model that would integrate both together into an output describing the pedal inputs of the car. Those three together are basically what adaptive cruise control is, combining normal cruise control with a collision avoidance feature to maintain separation or brake the vehicle

Finally, the model we created had issues with the braking range. Because we are using the centroid method, and the centre of mass of any membership function we use would never be able to reach zero, we decided the best fix would be to remove “None” from the braking pressure fuzzy set and instead use the crisp distance input to decide when the system should be active. So when an object is not detected, the braking pressure is ignored completely, circumventing this issue.
An alternative solution could have been tweaking the range, or better yet, using a different defuzzification method.