FRASER BARKER

1600196

1600196@uad.ac.uk

ASSESSMENT REPORT

MAT301

Contents

[Introduction 2](#_Toc6400494)

[AI Techniques 2](#_Toc6400495)

[Finite State Machine 2](#_Toc6400496)

[Fuzzy Logic State Machine 2](#_Toc6400497)

[Reasoning 2](#_Toc6400498)

[Computational Efficiency 2](#_Toc6400499)

[Ease of Coding 2](#_Toc6400500)

[Method 3](#_Toc6400501)

[Variables 3](#_Toc6400502)

[Input 3](#_Toc6400503)

[Output 3](#_Toc6400504)

[Test Data 4](#_Toc6400505)

[Results & Conclusions 5](#_Toc6400506)

[References 6](#_Toc6400507)

# Introduction

## AI Techniques

### Finite State Machine

A finite state machine was chosen as the first AI technique as it was easy to implement when considering the application would be moving a car sprite left/right depending on its distance away from a driving line and the speed it was moving at.

It’s also the easiest to compare against a fuzzy logic state machine is there is some overlap between the two techniques.

### Fuzzy Logic State Machine

A fuzzy logic state machine was chosen to compare against a finite state machine due to there being overlap in how both techniques operate.

The inference system was generated in MATLAB. This was then imported into the project using the FuzzyLite toolset. The result moves the car left/right depending on its distance from the driving line and its speed.

## Reasoning

As both techniques are state machines, the only differing factor is how the input data is used to calculate what direction the car should move in.

This made them an ideal comparison as they both function similarly.

## Computational Efficiency

### Finite State Machine

Largely made up of floats and an enum to depict what state the car is in. Aside from SFML variables to render the sprite it’s pretty simplistic.

### Fuzzy Logic State Machine

Similar to the finite state machine except the direction of the fuzzy car is dependent on its direction being calculated via FuzzyLite.

## Ease of Coding

Both systems were easy to set up. The trickiest part was getting the FuzzyLite library into the project which required building files via CMake.

# Method

## Variables

### Input

### Output

# Test Data

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

# Results & Conclusions

# References

Unknown. 2010. A Short Fuzzy Logic Tutorial. [ONLINE] Available at:

<http://cs.bilkent.edu.tr/~zeynep/files/short_fuzzy_logic_tutorial.pdf> [Accessed 17/04/2019]

Juan Rada-Vilela. 2017. FuzzyLite. [ONLINE] Available at:

<https://fuzzylite.com/> [Accessed 17/04/2019]

Ocornut. 2018. ImGui-SFML. [ONLINE] Available at:

<https://github.com/eliasdaler/imgui-sfml> [Accessed 17/04/2019]

Ocornut. 2019. ImGui. [ONLINE] Available at:

<https://github.com/ocornut/imgui> [Accessed 17/04/2019]

Bahi. 2011. Cartoon Car- Mini. [ONLINE] Available at:

<https://opengameart.org/content/cartoon-car-mini> [Accessed 17/04/2019]

Laurent Gomila. 2019. SFML. [ONLINE] Available at:

<https://www.sfml-dev.org/learn.php> [Accessed 17/04/2019]

MathWorks. 2019. MATLAB. [ONLINE] Available at:

<https://uk.mathworks.com/products/matlab.html> [Accessed 17/04/2019]

KitWare. 2019. CMake. [ONLINE] Available at:

<https://cmake.org/> [Accessed 17/04/2019]