Department of Computer Science & Applied Physics

B.Sc. Software Development
Artificial Intelligence Assignment 2023

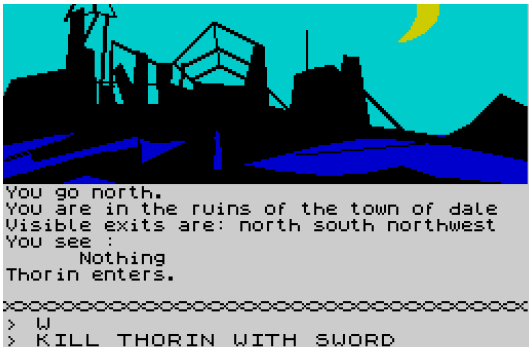
ASSIGNMENT DESCRIPTION & SCHEDULE

Interacting with AI-Controlled Characters in a Text Adventure Game

***Note*: This assignment will constitute 50% of the total marks for this module.**

1. Overview

Text adventures or Interactive fiction (IF) are a game genre where a story is conveyed to a player through passages of text, revealed in response to typed instructions. The game normally consists of a set of locations that the user has to navigate through. Navigation is typically implemented by the user typing a direction, e.g. N/S/E/W/NE/SW. Adventure games were *de rigueur* in the 1980s when home computing, using ZX Spectrums and Commodore 64s, became popular.



In order to play the game, a certain set of nouns and verbs should be supported. For example, using the verb "LOOK" might output a description of the current environment. "WAIT" might cause time to move forward. Typically however, verbs and nouns are used in adventure games to create primitive sentences. For example, to greet a character called "innkeeper", you might type "TELL innkeeper POUR ale". The command "KILL goblin WITH sword" might begin an altercation... Aside from characters, locations might also contain inanimate objects such as lances, swords, armour etc. A primitive sentence such as "GET sword" might add a sword to a character's inventory of objects.

You are required to create an AI controlled text adventure game containing the set of features listed below using a suite of stubs provided on Moodle. The purpose of this assignment is not to programme a highly polished game, but to *design a fuzzy logic system and neural network(s) that can control the enemy characters in the game* and make them act in a pseudo-intelligent manner. As such, much of the programming will be declarative (fuzzy control language) or AI design considerations, e.g. neural network topology and training data.

2. Minimum Requirements

You are required to create a text adventure game that challenges a player to complete some kind of a quest. The game should end, with suitable celebration and fanfare, when the appropriate player action fulfills the quest. The game should:

- Consist of a minimum of ***5 locations***. The locations should be connected and navigable by typing the ordinal points of a compass as the direction to travel in. Upon arrival in a new location, a picture and some narrative text should describe the location and list the characters currently situated there.

- Contain at least two different AI-controlled ***enemy characters***, one controlled by fuzzy logic and the other by a neural network. These characters should be free to roam though the locations, i.e. in a separate thread and may or may not attack you.

- Contain at least a sword, some food and other items that may be picked up and used. You can add any type of items that you want to a location.

- Provide a small ***grammar*** (a set of verbs and nouns) that allows a player (also a type of character) to interact with other characters. At a minimum you should have LOOK, GET, FIGHT, EAT and TELL verbs. ***Note that the verbs map to methods and the nouns to characters or items.***

- The fuzzy logic should be encapsulated in a FCL file and placed in the ***./resources/fuzzy*** directory. You should consider carefully the following when designing the fuzzy logic components of the system:

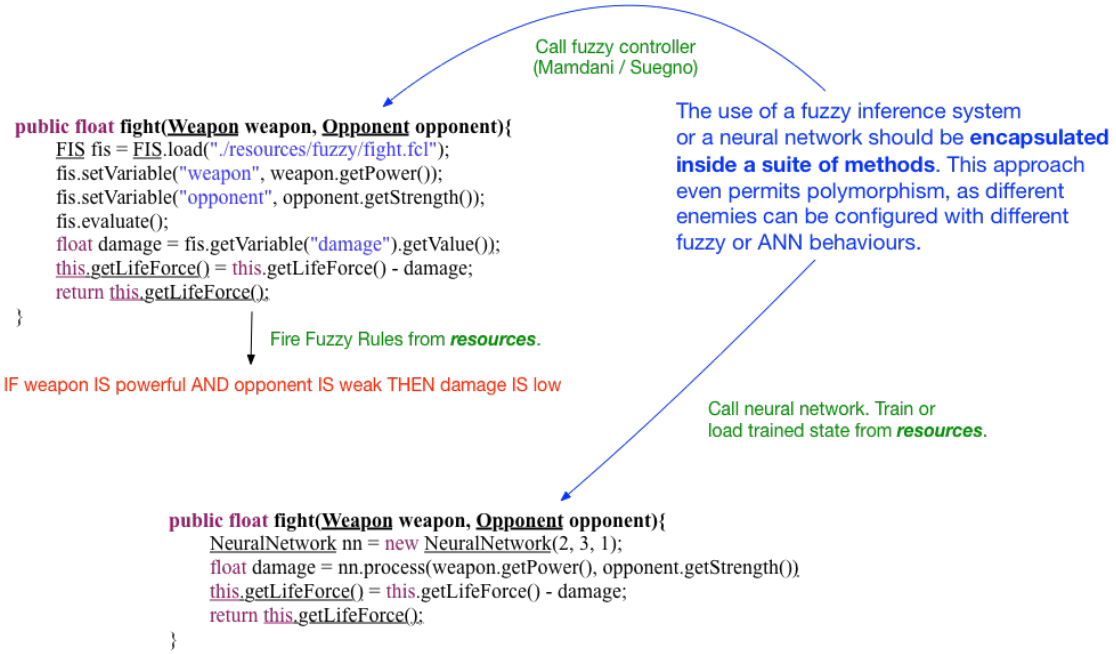
    1. **Linguistic variables** and Universe of Discourse
    2. **Fuzzy Sets** and Membership Functions
    3. **Defuzzifiers**, e.g. COG, COGS, LM, MM, RM.
    4. Membership Functions for output variables, e.g. **Mamdani** or **Suegno** (singleton values)
    5. **Fuzzy Rules**. All fuzzy sets must be covered by at least one fuzzy rule.

    You must ***comment the FCL with your rationale*** for each of the design decisions that you make. Note that you can include more than one function block per FCL file.

- The neural network should be trained, in a reasonable amount of time (< 1 minute), when the game starts up and have a **fully documented topology** presented in the README. You are free to use the either the **Aicme4J** neural network from Moodle or the **Encog** neural network that is also available on Moodle. Do not use any other 3rd party implementation. All the training and validation resources for the neural network should be placed in the ***./resources/neural*** directory.

You should view the application of the fuzzy logic or neural network as an "intelligent" implementation of a standard method. Consider the example method ***fight()*** shown below. The method parameters can easily be used as the input to either a fuzzy inference system or a neural network. The output from the fuzzy controller or neural network can be used as a method return type.

Note: you are free to asset-strip any of the resources, including labs and source code, available on the Moodle page for this module.



3. Deployment and Delivery

Please **read the following carefully** and pay particular attention to the files that you are required to submit and those that you should not include with your assignment:

- ***The project must be submitted by midnight on Sunday 9th April 2023***. Before submitting the assignment, you should review and test it from a command prompt on ***a different computer*** to the one that you used to program the project.
- The project must be submitted as a Zip archive ***(not a 7z, rar or WinRar file)*** using the Moodle upload utility. You can find the area to upload the project under the *"Interacting with AI-Controlled Characters in a Text Adventure Game - (50%) Assignment Upload"* link on Moodle. Only the contents of the submitted Zip will be considered. ***Do not add comments to the Moodle assignment upload form.***
- The name of the Zip archive should be <*id*>.zip where <*id*> is your ATU student number.

Do NOT submit the assignment as an Eclipse project or submit any text files or other resources. If you do, you will lose marks. You will also lose marks if you hard-code any environmental variables in your project like system paths and file names.

- The Zip archive should have the structure shown below.

Marks	Category
ai-text-adventure.jar	A Java archive containing your API and runner class with a ***main()*** method. You can create the JAR file using the following command from inside the "bin" folder of the Eclipse project: **jar –cf ai-text-adventure.jar *** The application should be executable from a command line as follows: **java –cp ./ai-text-adventure.jar ie.atu.sw.ai.Runner**
src	A directory that contains the packaged **source code** for your application.
README.pdf	A PDF file detailing the **design and main features** of your application. Marks will only be given for features that are described in the README.

4. Marking Scheme

Marks for the project will be applied using the following criteria:

Marks	Category
(10%)	**Packaging & distribution.** All or nothing. The application must be structured correctly and execute without any manual intervention as described above.
(10%)	**Threads.** The player and enemy sprites should execute in separate threads.
(30%)	**Fuzzy Logic (10% for Each of the following):** • Fuzzy sets and membership functions with comments in README. • Fuzzy rules (all sets should be covered by at least one rule) with comments in README • Integration with game characters.
(30%)	**Neural Network (10% for Each of the following):** • Network topology, normalization and accompanying rationale in README. • Training and validation data with comments in README. • Integration with game characters.
(20%)	**Semantic Network and Grammar Set (10% for Each of the following):** • Fully commented design in the README and implementation of a state space for the text adventure with a minimum of 5 locations. • Implementation of LOOK, GET, FIGHT, EAT and TELL verbs at a minimum.

You should treat this assignment as a project specification. Any deviation from the requirements will result in some or all marks not being earned for a category. Each of the categories above will be scored using the following criteria:

Range	Expectation
0–39%	***Not delivering*** on basic expectations. The submission does not meet the minimum requirements.
40-60%	Meets ***basic*** expectations. The minimum requirements are met in whole or in part.
61–74%	Tending to exceed expectations. A ***high-quality*** assignment with additional functionality added.
75-89%	Exceeding expectations and ***very high-quality***. Any mark over 70% requires evidence of ***independent learning*** and cross-fertilization of ideas, i.e. your project must implement a set of features using advanced concepts not covered in depth during the module.
90-100%	***Exemplary***. Your assignment can be used as a teaching aid with little or no editing.