KE5207 COMPUTATIONAL INTELLIGENCE II

USER MANUAL FOR FUZZY LUNAR LANDER

Guide on using fuzzy lunar lander program

SUBMISSION DATE: 21 OCTOBER 2018

TEAM: FAMOUS FIVE
SIDDHARTH PANDEY
PRANSHU RANJAN SINGH
EDUARD ANTHONY CHAI
NYON YAN ZHENG
TAN KOK KENG

INSTITUTE OF SYSTEMS SCIENCE
NATIONAL UNIVERSITY OF SINGAPORE

Contents

Prerequisites

The game is written in Java programming language, thus a suitable Java runtime must be installed on the system to play it. Java Runtime Environment 8 can be downloaded from https://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html. The game is tested using Java 8 runtime, but should work on newer runtime environments also.

Other than Java 8 there are no other perquisites.

Checking Java version

Inside the command prompt or Linux terminal type the following command

```
>java -version
```

The output should be similar to this

```
java version "10.0.2" 2018-07-17
Java(TM) SE Runtime Environment 18.3 (build 10.0.2+13)
Java HotSpot(TM) 64-Bit Server VM 18.3 (build 10.0.2+13, mixed mode)
```

Make sure the Java version 8 or above is installed.

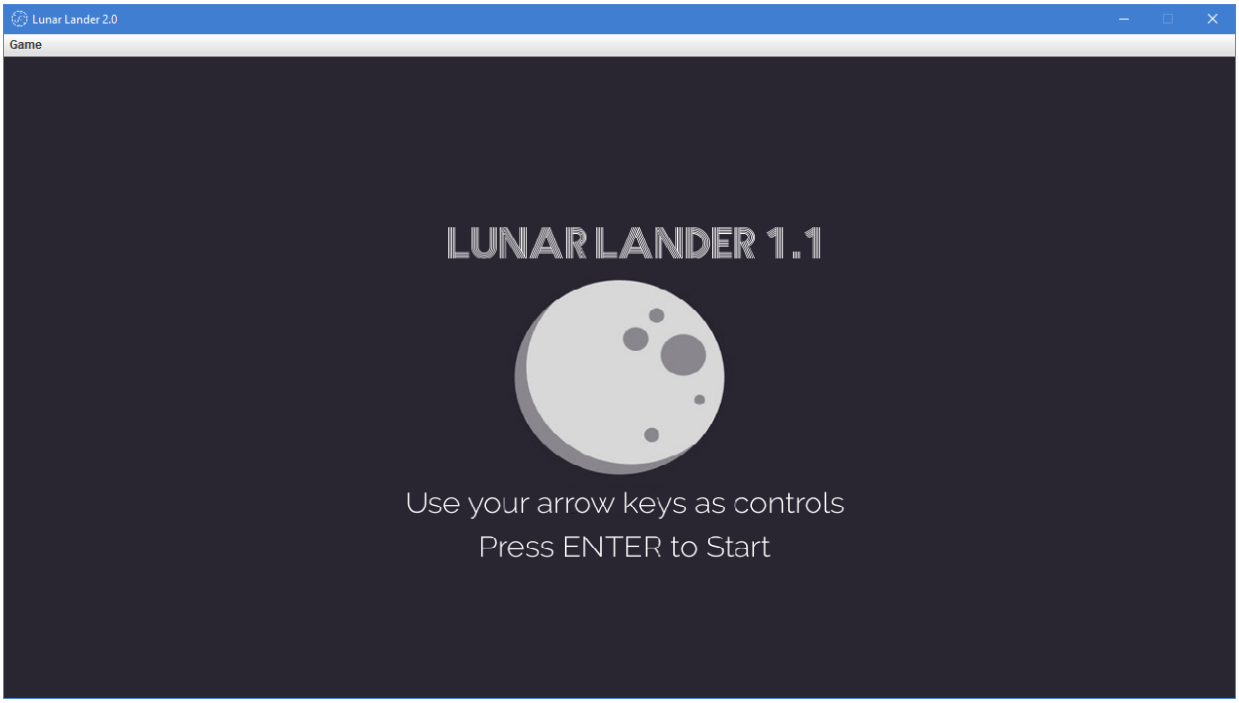
Starting the Game

Double Click on lunarlander_v2.jar file to start the game. If Java is installed correctly, game's welcome screen should appear.

Running Command Line

Navigate to the location where the jar file is. And the run the following command:

```
>java -jar lunarlander_v2.jar
```

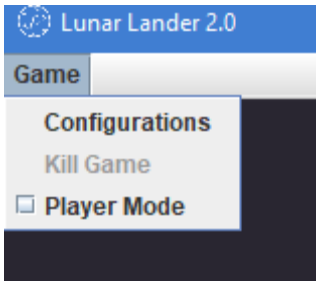


Playing the game

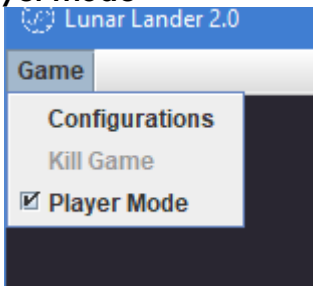
Switching to Player Mode

By default the game start with fuzzy bot controlled mode, which pilots the rocket. To be able to control the rocket yourself, the game needs to be switched to the player mode.

Click on Game menu -> Select Player Mode

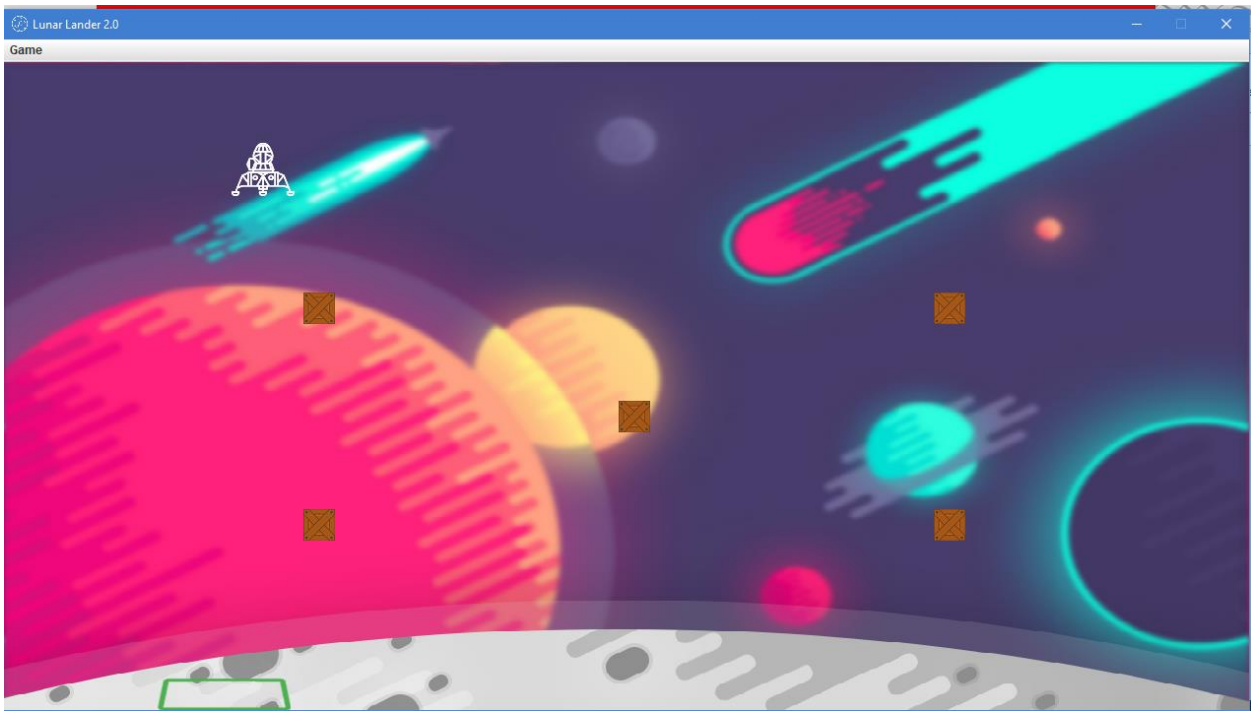


Currently in fuzzy bot mode



In player mode

Press enter to start the game



The game will start with a random position of Rocket and Platform and a fixed arrangement of obstacles. Land the space ship safely onto the green platform to win the game.

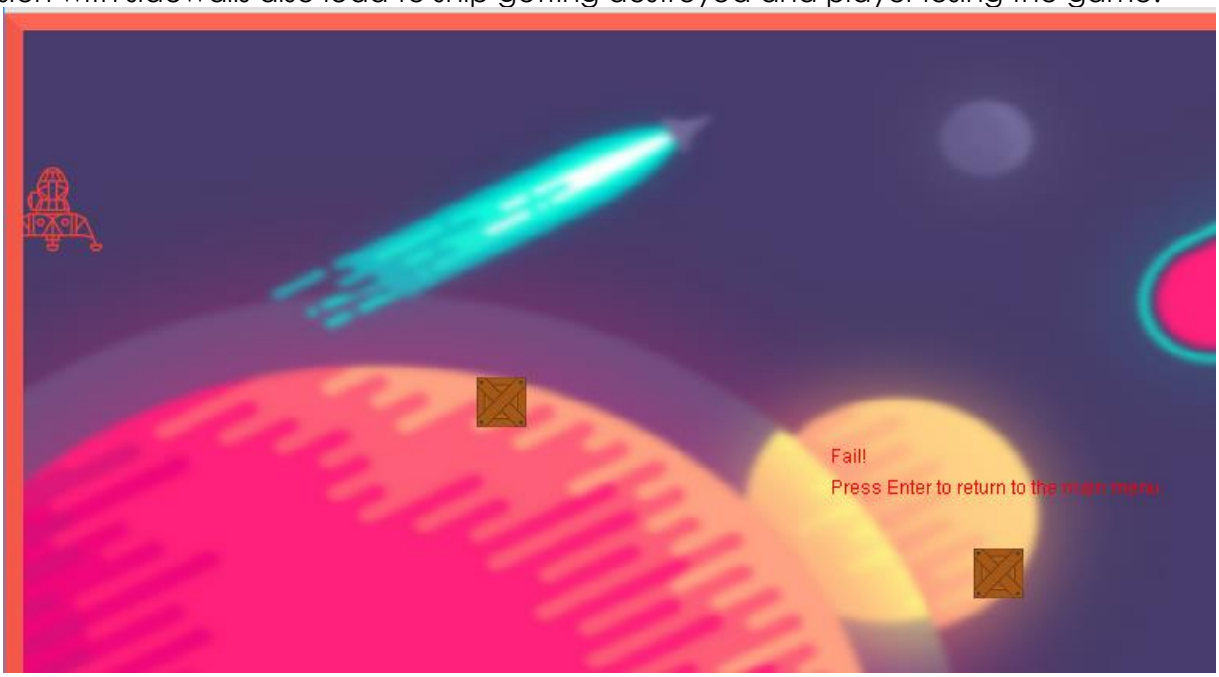
Use the keyboard arrows to navigate the rocket safely to landing platform.

Avoiding Obstacle

Collision with obstacles, destroys the ship, and the player loses the game

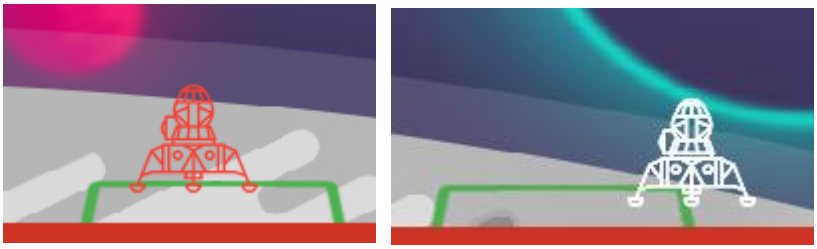
Avoiding Sidewalls

Collision with sidewalls also lead to ship getting destroyed and player losing the game.



Landing on Platform Erroneously

Player also loses if he lands on the platform with high speed or if a part of ship is outside the platform.



Winning the Game

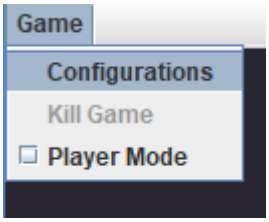
To win the game player must navigate through obstacles, avoid walls and land softly on the platform.

Switching to Fuzzy Bot Mode

Uncheck the Player Mode, in the game menu to switch to bot mode.

Game Configuration Menu

Understanding the Menu



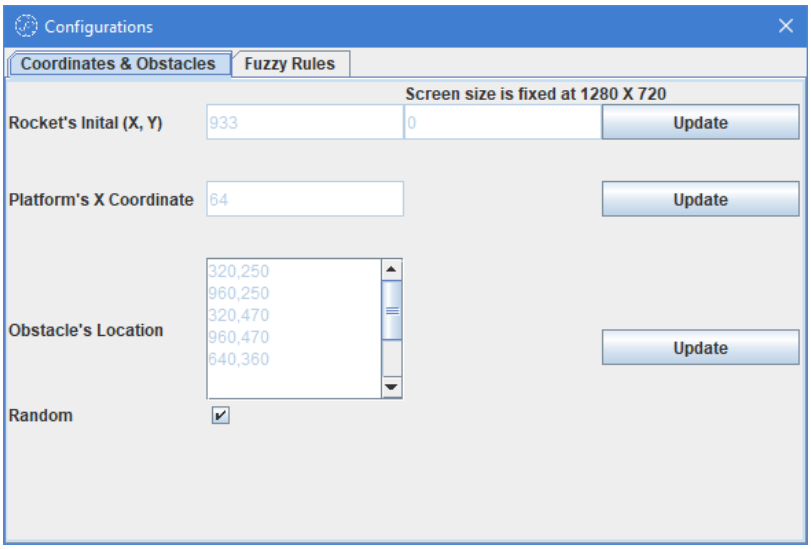
There are three menu items in the game menu

1. Configurations: Lets the player tweak the settings of the game. Discussed in next section. This item is disabled when the player is playing.
2. Kill Game: Instantly destroys the ship and causes game over. Helpful if fuzzy bot gets stuck in loop. This item is disabled when on the game welcome screen.
3. Player Mode: Allows to switch between player mode and bot mode. This item is disabled when the player is playing.

Configuration Options

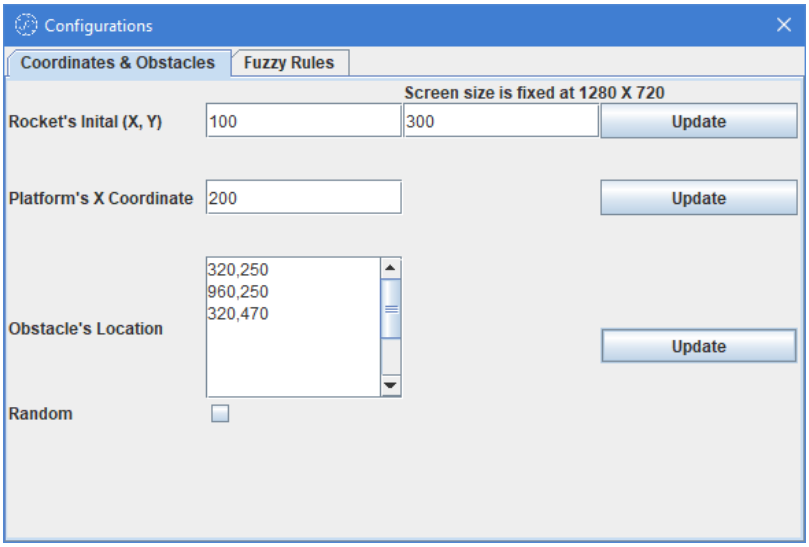
Many of the game components can be tweaked to test performance of fuzzy bot and make the game more challenging for both the human and AI player.
We allow following configuration changes:

1. Rocket's spawning position
2. Platform's X-coordinate
3. Number of obstacles
4. Location of each obstacle
5. Rule set



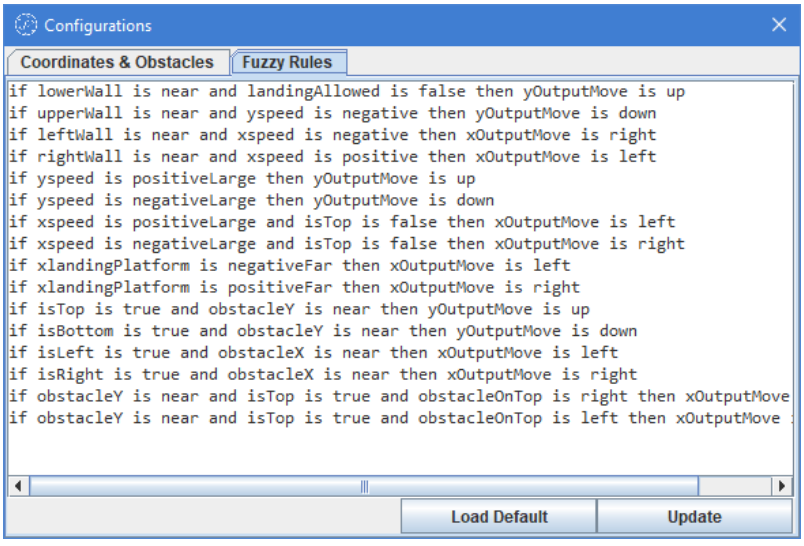
Un-check the Random checkbox to be able to edit the values. After every change, corresponding update button needs to be clicked to save the changes. Finally the window can be closed to start the game with the new settings.

Key in values of coordinates to change rockets spawning position and platform X coordinate.
To change the number of obstacles, modify the obstacle's location.



The number of (X, Y) pairs define the number of obstacles the system will spawn. The value (X, Y) is the coordinate of centre point of the obstacle.

Remember to click update to save the changes in corresponding configuration.



The other tab allows to change the fuzzy rules. Rules can be added/removed.
Click on "Load Default" to restore the rules to the default state.

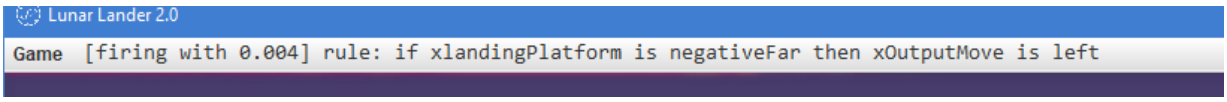
For available variables and linguistic terms refer to project report.
New rules must follow the same syntax as the ones already loaded.
Refer to FuzzyLite documentation https://fuzzylite.com/fll-fld/.

Understanding Game Logs

Game logs are very helpful to understand which rules are firing for given instance, and what is move the fuzzy bot is playing.

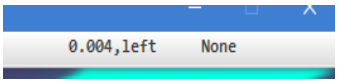
Logs are displayed automatically on game menu bar, when fuzzy bot is playing.

Rule Firing Log



The user can view which rule is being currently fired and firing strength of the rule. This helps us understand how the bot is behaving in specific situations.

Final Decision Log



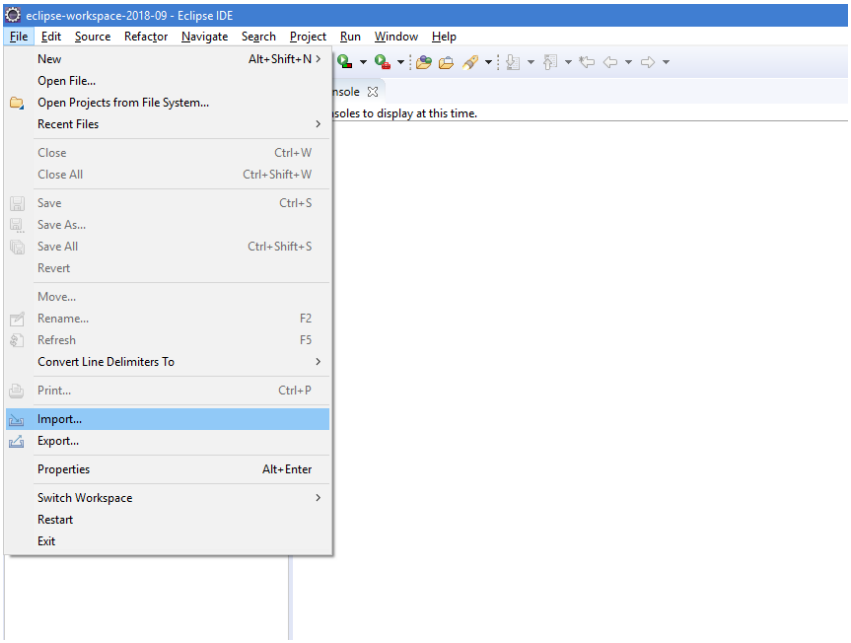
After the rules are fired, the moves with maximum activation in Left/Right and Up/Down are selected. The ship can move in both axis simultaneously, allowing for diagonal movements. The decision log have two section, one for each axis, displaying the action it will take. In this case bot will move left, and not perform any action in y direction.

Modifying the Source Code

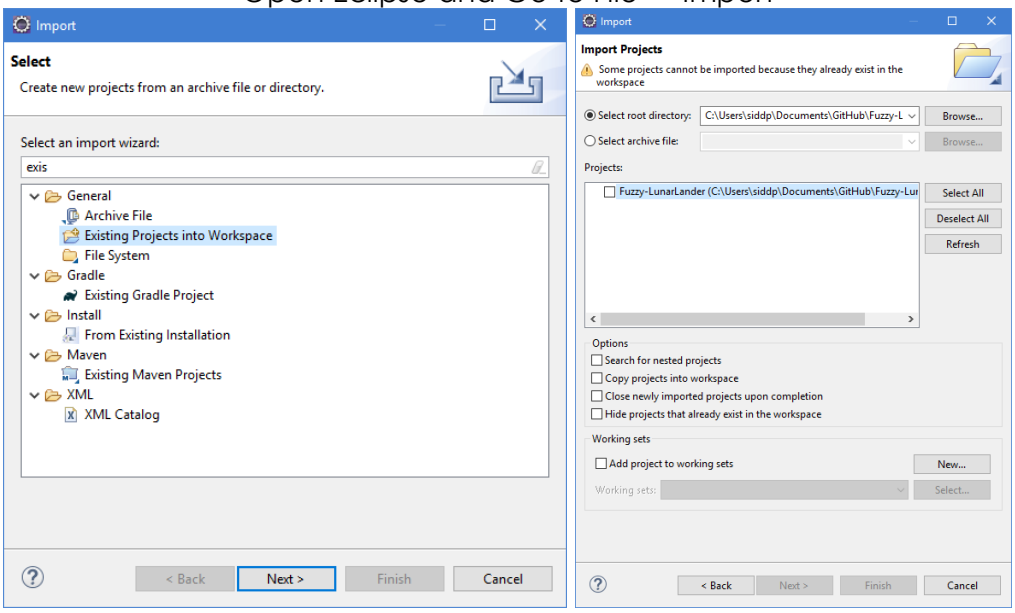
While we have put in lot of effort to make most of the game components configurable, the membership functions and linguistic variables cannot be modified from user interface. In case the user wants to make changes in these, he/she must make changes in the source code and recompile the game. Below we describe how to import the project in Eclipse IDE, to get started on modifying the source.

Extract the attached source code, or clone from the GitHub repository link https://github.com/sidd-pandey/Fuzzy-LunarLander/.

Importing to Eclipse

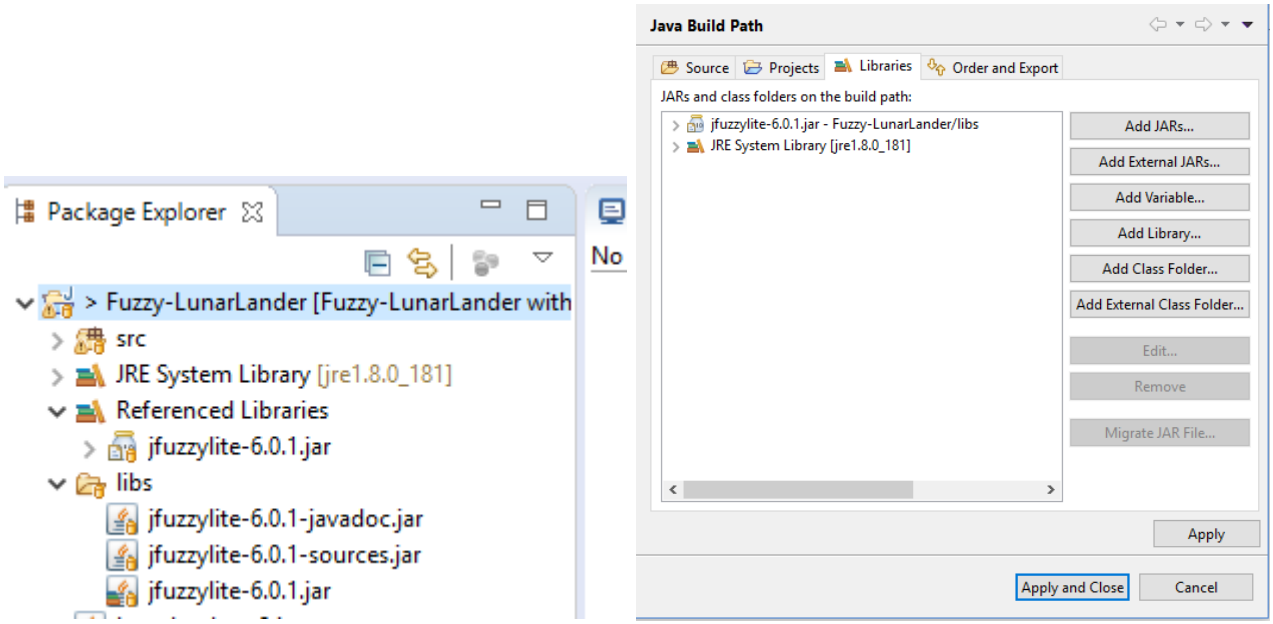


Open Eclipse and Go to File -> Import



Select Import Existing Projects into Workspapce, and provide location of Fuzzy-LunarLander source foder, clikc finish to complete.

Checking Class Path



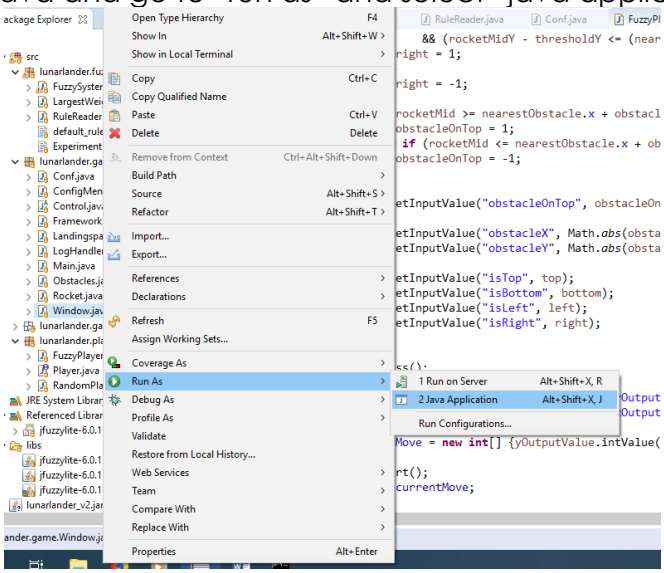
We will require the jfuzzylite jar file in the class path to successfully compile the source code. Make sure they are present, by checking if it listed in Reference Library dropdown of project explorer. If not modify the Java Build Path, and click add Jar and point to location of JFuzzyLite jar. Refer to documentation and tutorial on adding external jar files in class path using Eclipse.

Important Classes & Files

- **FuzzySystem.java** hosts all the input and output variable of fuzzy inference engine. To make changes in membership function, head down to Line 53.
- **FuuzyPlayer.java** contains program logic to transfer the current game state to inference engine and get the next move to play,
- **Windows.java** is the entry point to program
- **default_rules.txt** contains the set of rules to load in the engine.

Running the Game

Right click on Windows.java and go to "run as" and select "java application" to run the game.



Concluding Remarks

Lunar Lander is one of the classic Atari arcade games reminiscent of "golden age of arcade video games". The very aspect of navigating the game environment delicately and avoiding charging towards goal greedily because the ship's speed increase exponentially, introduces an element of fuzziness in control tactics of the ship. We enjoyed building a fuzzy inference based automated bot to play the game. We also realized that there is no golden set of rules that makes bot navigate in all environment settings, and that there is sometimes need to tweak membership function a bit, to successfully clear the level.