Game 4

Vector Field Pathfinding

Vector Field Pathfinding ~ Introduction (1)

- Pathfinding can be considered as a staple AI algorithm, used extensively throughout games.
- In today's market, one of the most frequently used pathfinding algorithms is the **A* Pathfinding Algorithm**, which works by calculating the shortest route when given a graph to navigate. However, this algorithm is **not always optimal** in all scenarios [1].
- The A* Pathfinding Algorithm may not always present the optimal solution in the following scenarios:
  - An abnormal number of agents who are concurrently attempting to arrive to the same destination.
  - A highly dynamic Environment.

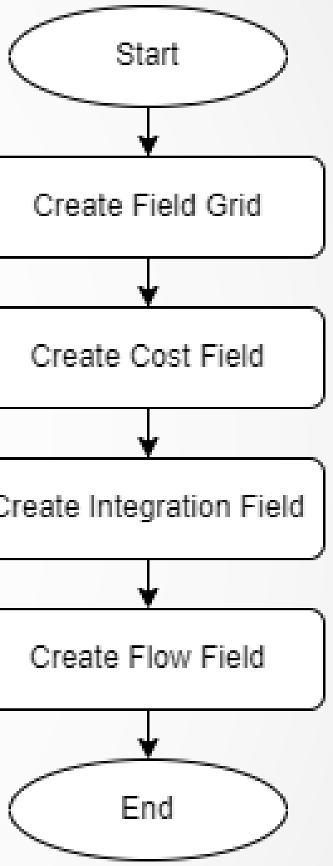
Vector Field Pathfinding ~ Introduction (2)

- Enter **Vector Field Pathfinding**, a pathfinding algorithm which addresses the previously mentioned limitations of the A* Pathfinding Algorithm [1].
- As the name suggests, this algorithm utilises vectors to guide agents to the required destination.
- Moreover, this pathfinding algorithm makes use of a Grid of Cells, whereby each cell contains a vector that points in the direction of the neighbour which is nearest to the goal / destination.
- Once the Vector Field is calculated, an agent situated on a cell would utilise the vector of that cell in the calculation of the agent's final velocity to reach the target.

Vector Field Pathfinding ~ AI Explanation (1)

- Implementation of the Vector Field Pathfinding requires the construction of the Flow Field.
- Moreover, calculation of the such Field can be partitioned into four processes, as can be seen in the following diagram:

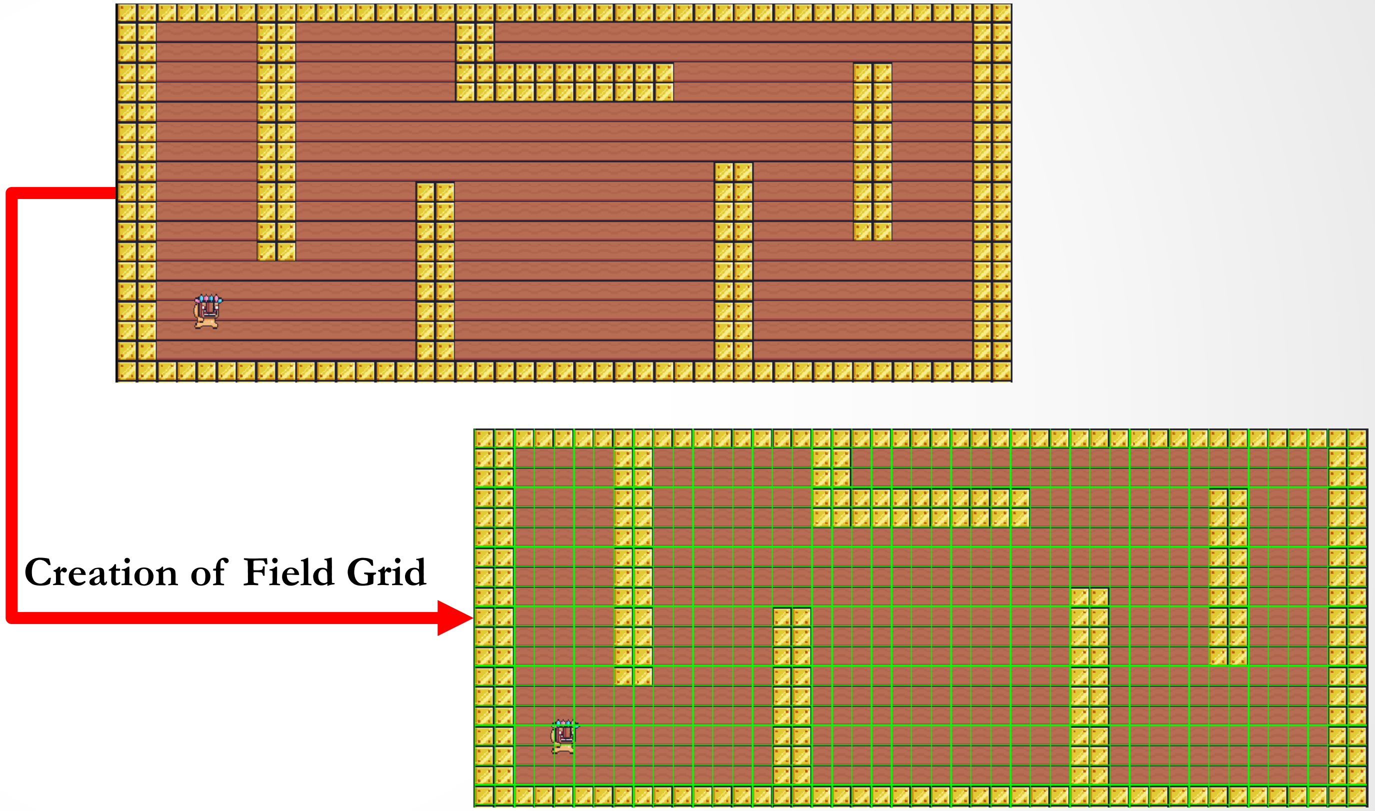
1. **Creation of Field Grid**
2. **Creation of Cost Field**
3. **Creation of Integration Field**
4. **Creation of Flow Field**



Vector Field Pathfinding ~ AI Explanation (2)

1. **Creation of Field Grid:**

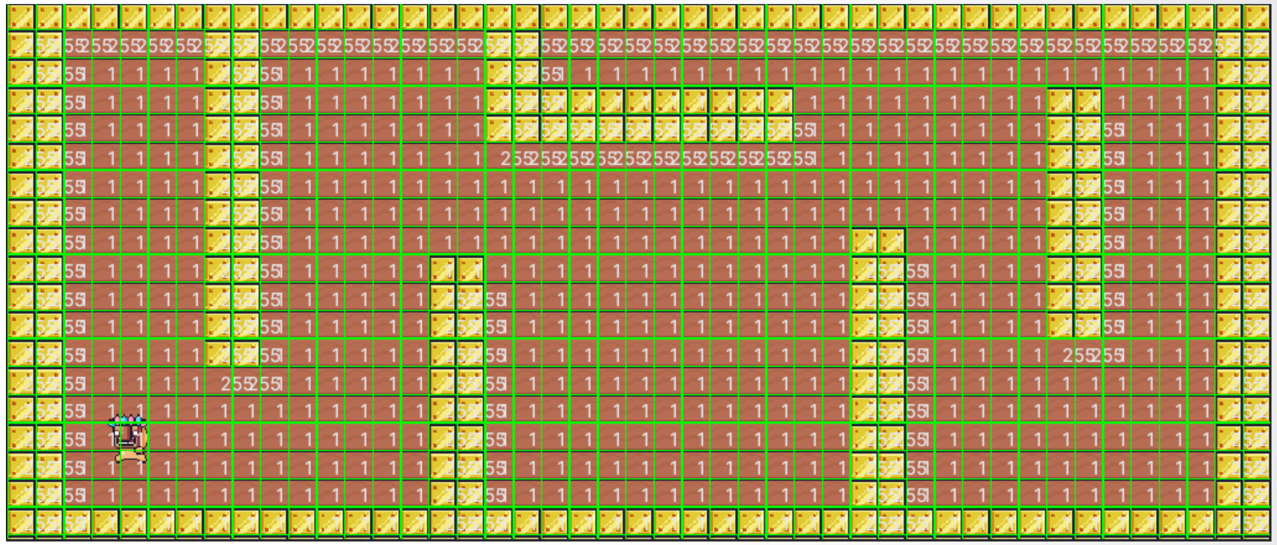
- What is the Field Grid ?
- The Field Grid determines the area over which the Flow Field will apply, in other words it determines the area where AI agents are able to move in.



Vector Field Pathfinding ~ AI Explanation (3)

2. Creation of Cost Field:

- What is the Cost Field?
- By observing the terrain which collides with the Field Grid, depending on the type of terrain, a cost is issued to each cell in the Field Grid, by default cells are given a cost of 1, and walls are given a cost of 255 [1].

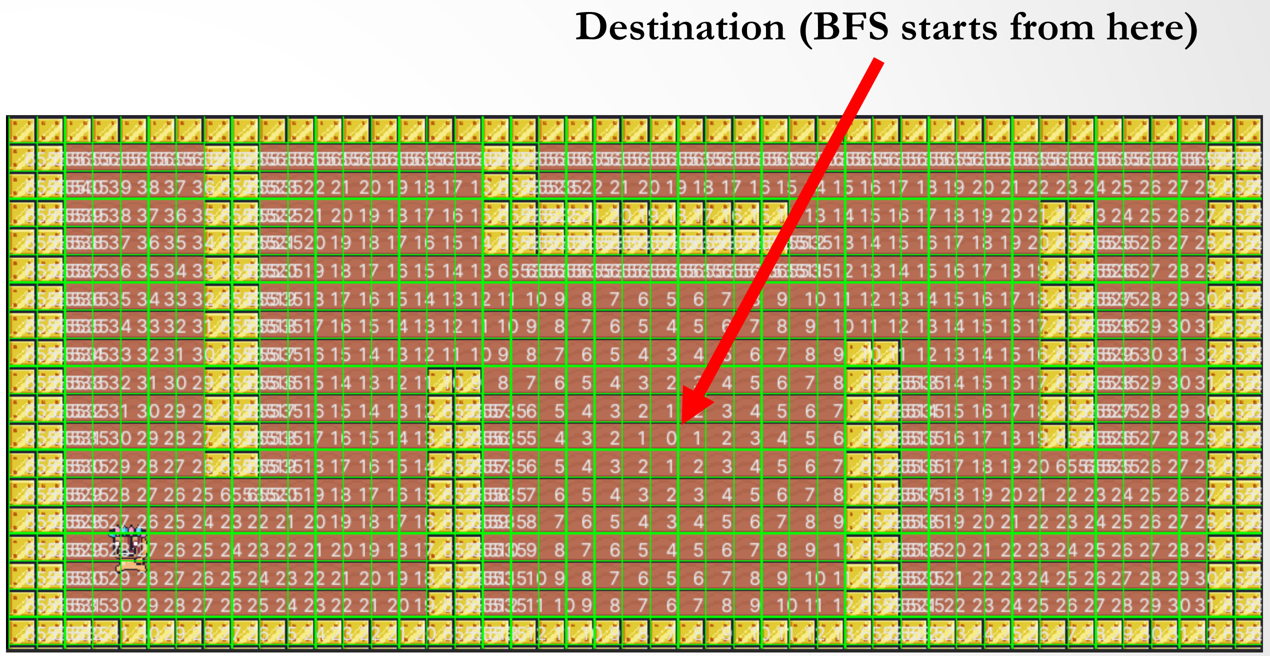


Creation of Cost Field

Vector Field Pathfinding ~ AI Explanation (4)

3. **Creation of Integration Field:**

- What is the Integration Field?
- Most of the computation, when creating a Vector Flow Field falls in this stage.
- To explain it briefly; a destination Cell is outlined (Cost of 0), and a queue of Cells is created.
- Next, a Breadth First Search is initiated from the destination Cell, to facilitate the calculation of each cell's best cost to reach the goal [2].

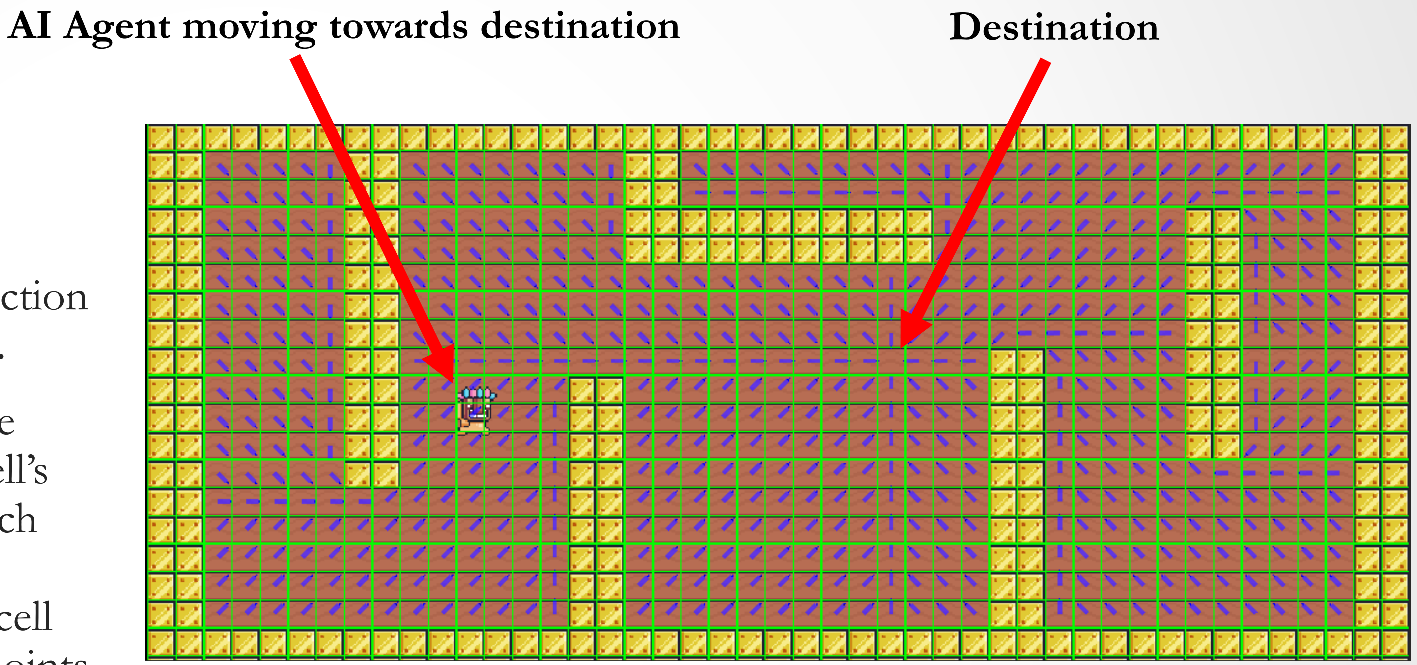


Creation of Integration Field

Vector Field Pathfinding ~ AI Explanation (5)

4. **Creation of Flow Field:**

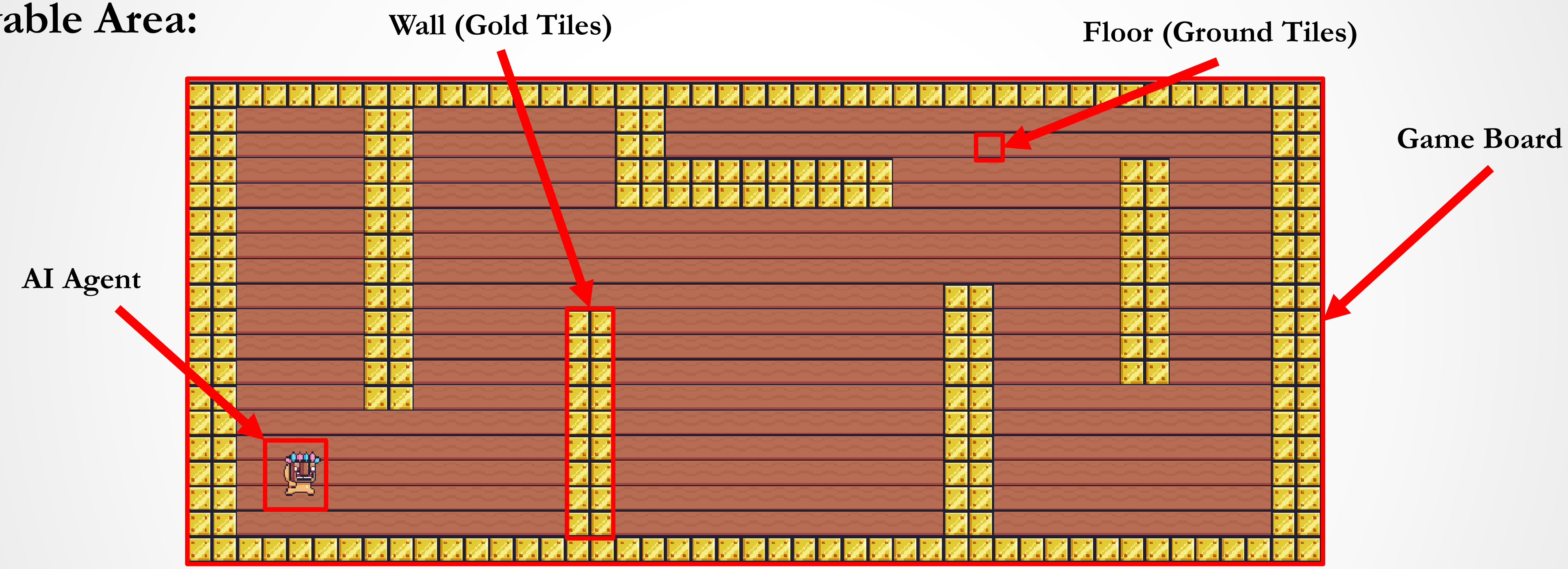
- What is the Flow Field?
- The Flow Field will determine the direction of the vectors in each cell of the Field.
- Explaining it briefly; for each cell in the field, the algorithm, will observe the cell's valid neighbours, and depending on each neighbour's best cost calculated in the previous stage (Integration Field), the cell will be initialised with a vector which points to the neighbour with the smallest (best) cost [1].



Creation of Flow Field

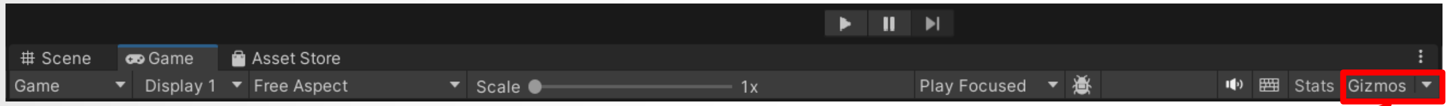
Vector Field Pathfinding ~ Mini-Game Implementation (1)

Playable Area:



Vector Field Pathfinding ~ Mini-Game Implementation (2)

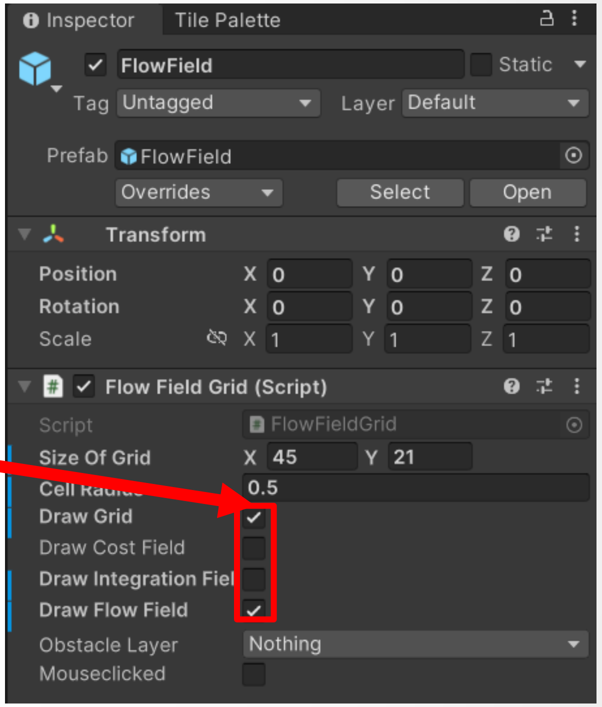
Developer Interface:



Gizmos Button

- As can be seen in the following images, using the Gizmos button, developers, can enable / disable the visibility of the indicated gizmos, for easier debugging (Gizmos button needs to be enabled to view the different fields).
- Developers, can choose to view the different fields outlined in the slides above, through the different togglable buttons, on the Flow Field Game Object.

Different Fields Togglable buttons


Inspector | Tile Palette

FlowField ✓ — Static

Tag: Untagged — Layer: Default

Prefab: FlowField

Overrides — Select — Open

Transform

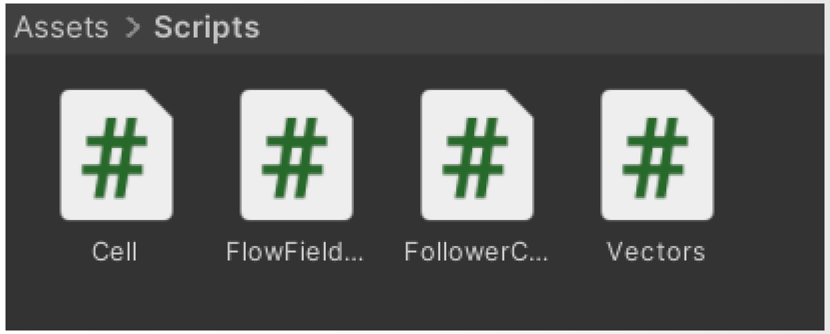
	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	1	1	1

Flow Field Grid (Script)

Script	FlowFieldGrid
Size Of Grid	X 45 Y 21
Cell Radius	0.5
Draw Grid	✓
Draw Cost Field	
Draw Integration Fiel	
Draw Flow Field	✓
Obstacle Layer	Nothing
Mouseclicked	

Vector Field Pathfinding ~ Mini-Game Implementation (3)

- Implementation of the Mini Game was inspired from [2-3], and sprites used to create the game were retrieved from [4].
- **The Game is Composed of the following scripts:**
  - **Cell** script – This script is being used as a data structure, to initialise each individual cell which compose the field grid. The cells can be seen in slide 5 as a green box.
  - **FlowField** script – This script is being used to create the Field Grid, Cost Field, Integration Field and the Flow Field (contains most of the computation for the pathfinding algorithm).
  - **Vectors** script – This script is being used as a data structure, to initialise each cell with a vector, as to indicate the direction of movement.
  - **FollowerController** script – This script is being used to move the agent / follower from the starting point to the specified end point. The path taken is taken via the Cell Vectors, which indicate to the agent / follower, the direction of movement.



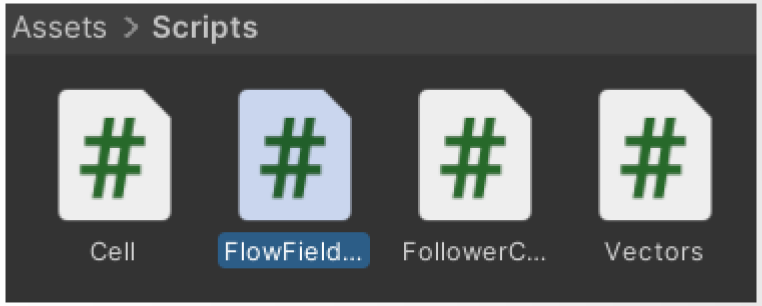
Vector Field Pathfinding ~ Exercise (1)

Now Its your turn to Code! – Let's implement the Vector Field Pathfinding Algorithm ☺

1. Navigate to the Assets> Scripts folder, and open the **FlowField** script
2. In the Flow Field script find the **CreateFlowField()** method
3. Utilise the following Pseudocode to populate this method (CreateFlowField() method)

Pseudocode:

```
1. Loop through all the cells in the cellGrid
2. Retrieve the Cell Neighbors and storing them in a list of Cells
   • (Hint: use the GetNeighboringCells(currentCell.gridIndex, Vectors.allDirections) to retrieve
     neighbors)
3. Store the current Cell's best Cost in a variable
```

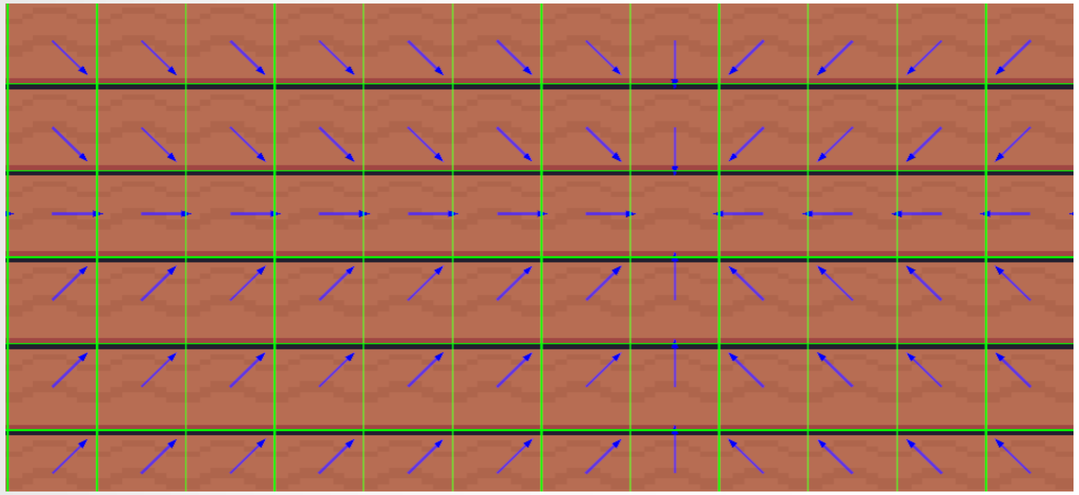


Vector Field Pathfinding ~ Exercise (2)

Pseudocode Continue…

4. Looping through the list of neighbors, obtained in 2.
5. Inside the secondary loop check whether the neighbor's best cost is better than the cell best cost
   - (Hint: use the variable obtained in 3.)
6. If the condition in 5. is satisfied then do the following:
   1) Set the bestCost variable to the neighbor's bestCost
   2) Create a new Vector2 variable called vector and store the difference between the neighbor.gridIndex and currentCell.gridIndex
   3) Set the currentCell.bestDirection to new Vectors(vector.x,vector.y), where Vectors is a different Class to denote direction, and vectors is the variable declared in 2)

Vector Field Pathfinding ~ Conclusion



- Although Vector Field Pathfinding can be quite tedious to implement when compared to Pattern Movement, it provides a variety of advantages, such as being dynamic, provides wall avoidance, as well as being able to handle multiple agents concurrently.
- Notwithstanding, in some cases, A* Pathfinding provides a better solution, whilst in others, Vector Field Pathfinding provides a better result.
- Through this PowerPoint, the Student would be able to know and identify ways of how Vector Field Pathfinding can be implemented, as well as being able to differentiate between different Pathfinding algorithms.

Vector Field Pathfinding ~ References

[1] – L. Erkenbranch, "Flow Field Pathfinding" 2013 [Online]. Available: https://leifnode.com/2013/12/flow-field-pathfinding/ [Accessed: 18-Mar-2023]

[2] – Prof. A. Dingli, ICS2211: "Level3_Pathfinding" [Online]. Available: https://www.um.edu.mt/vle/pluginfile.php/1108327/mod_resource/content/1/Level3_PathFinding.pdf [Accessed: 18-Mar-2023]

[3] – Turbo Makes Games, Tutorial - Flow Field Pathfinding in Unity, 2020 [Online video]. Available: https://www.youtube.com/watch?v=tSe6ZqDKB0Y [Accessed: 18-Mar-2023]

[4] – Pixel Frog, "Unity Asset Store: Pixel Adventure 1" 2019 [Online]. Available: https://assetstore.unity.com/packages/2d/characters/pixel-adventure-1-155360 [Accessed: 18-Mar-2023]