Bachelor thesis in the Computer Science and Media degree programme

Tactical Game AI with shared Knowledge based on Influence Maps

submitted by

FRED NEWTON, AKDOGAN

at the Stuttgart Media University on August 30, 2021

to obtain the academic degree of Bachelor of Science (B.Sc)

First examiner: Prof. Dr. Stefan Radicke

Second examiner: Prof. Dr. Joachim Charzinski

Abstract

Honorary Declaration

Hiermit versichere ich, Fred Newton Akdogan, ehrenwörtlich, dass ich die vorliegende Bachelorarbeit (bzw. Masterarbeit) mit dem Titel: "Tactical Game AI with Shared Knowledge based on Influence Maps" selbstständig und ohne fremde Hilfe verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen wurden, sind in jedem Fall unter Angabe der Quelle kenntlich gemacht. Die Arbeit ist noch nicht veröffentlicht oder in anderer Form als Prüfungsleistung vorgelegt worden. Ich habe die Bedeutung der ehrenwörtlichen Versicherung und die prüfungsrechtlichen Folgen (§26 Abs. 2 Bachelor-SPO (6 Semester), § 24 Abs. 2 Bachelor-SPO (7 Semester), § 23 Abs. 2 Master-SPO (3 Semester) bzw. § 19 Abs. 2 Master-SPO (4 Semester und berufsbegleitend) der HdM) einer unrichtigen oder unvollständigen ehrenwörtlichen Versicherung zur Kenntnis genommen.

Signature	Date

Contents

Chapter 1

Introduction

1.1 Motivation

During his studies, Mr Akdogan always wondered how the game AI shares its knowledge. Because if the artificial intelligence (AI) always knows everything, it would not be beneficial for the player and you want to give the player a good experience. While looking for a topic for his bachelor thesis, he came across a game developers conference (GDC) video from [Brewer and Graham, 2020] that talks about the representation of information and also about influence maps for the AI. This raised the question of how much influence shared knowledge has among AI as opposed to agents not sharing their knowledge with each other.

1.2 Scientific question

How big is the difference between the AI agents when they share their influence map or each agent uses its own influence map?

1.3 Structure of the Thesis

Chapter 2

Related work

In [Champandard, 2021] article discussed how influence map (IM) works in general. As well as the important part of giving an agent or squad a memory of the current influence range on the map.

Chapter 3

Theoretical background

3.1 Influence Map

All information is taken from the book AI for Games unless otherwise cited in this chapter [Mellington, 2020].

An IM is used to record the current balance of Military Influence at each position in a level. Many factors can affect military influence, such as the proximity of a unit, the proximity of a base, the length of time a unit has been last seen, the terrain, the current weather, the strength of a unit. Many factors have only a small influence. Based on how the abstract image is drawn across the map or level of influence, more marginal information can be communicated to the AI and tactical decisions can be made.

3.1.1 Simple Influence

The influence of a unit in an area consists of how much its influence is weighted. Assuming it is a Real Time Strategy game and there is a foot soldier unit and a tank. Normally, a tank has more lives, more damage and a longer range than a simple foot soldier. This means that a larger Influence value is taken and injected into the IM at the unit's position. If you take the strength of a unit, it decreases with increasing distance. So the further away you are from the unit, the less influence it has. A linear drop-off model can be used for this. A doubling of the distance results in a halving impact:

$$I_d = \frac{I_0}{1+d} \tag{3.1}$$

I_d is the influence at a given distance. d is the distance from the unit to the point and I_0 is the influence at the distance value 0 to the unit. It would also be possible to use a more sloping initial drop off, with a greater range of influence:

$$I_d = \frac{I_0}{\sqrt{1+d}} \tag{3.2}$$

It is also possible to use an equation that first flattens out and then falls sharply:

$$I_d = \frac{I_0}{(1+d)^2} \tag{3.3}$$

3.1.2 Calculating the Influence

For the IM, a large calculation is needed for each unit on the map for each possible position. The execution time would be $O(nm)$ and the memory is $O(m)$. m represent the number of possible positions in the game and n the number of units. With a linear drop-off curve, the influence is covered with a threshold value. In this way, small values are not unnecessarily stacked on top of each other in a larger range:

$$r = \frac{I_0}{I_t - 1} \tag{3.4}$$

Where I_t is the threshold value for the influence. Thus, the influence of each unit is only applied to the places that are within the given radius. This limits the calculation time to $O(nr)$ for the time and to $O(m)$ for the memory. r is the number of locations that are within the given radius.

3.1.3 Dealing with unknowns

Here, only the influence of units that can be seen in their radius is calculated for the unit. Thus, an aspect called fog-of-war (FOW) is built in. This is important for investigating whether the shared knowledge of units makes a difference. In this way, units also have a maximum distance they can see and can only build a personal IM based on the friendly or enemy units they can see and incorporate this into their decisions. This can be seen as a problem, as the AI cannot simply assert which unit can be in the FOW as humans can. Furthermore, it becomes important here to see how much influence shared knowledge has among a team and thus the FOW becomes smaller and the unit shares knowledge among itself instead of each unit interacting individually.

3.1.4 Influence Map Setup

All information in this section is quoted from the article [Champandard, 2021]. Otherwise they are cited as such.

For the IM, a 2-dimensional grid is stretched over the map and divided into a grid system. Then all areas that cannot be walked on, such as walls or similar, are excluded from the calculation or ignored. This promotes the unknown, because you cannot see through walls. After that, the cells that are the shortest distance to each agent are injected with their influence in the 2 dimensional grid. This means that these cells are always set to the influence value of the unit regardless of anything else. After setting the values of each agent in the

IM, a blur algorithm is applied as explained in the chapter 3.2. It is up to you which one to use. The important thing is to only apply this to cells that are accessible. This way the unknown is applied and the influence of an agent has to spread around obstacles and not just take the distance to them. The value from the blur algorithm is then multiplied by the decay to implement a decay of the influence on the range.

$$I_{xy} = b_{xy} * D \tag{3.5}$$

I_{xy} is the influence at the point x and y in the grid. This is equal to the blurred value b_{xy} at the point x and y from the algorithm multiplied by the decay value D.

- **Momentum**
- **Decay** Decay is for the decay of the influence value within an IM so a kind of fading memory is built up and the influence continues to decrease depending on how far it is from its point of origin.
- **Update Frequency** This parameter describes how often the influence is updated.

3.2 Blur

For the calculation of the influence in maps with small corridors and narrow spaces Champandard [2021] a blur algorithm can be used as well as flooding functions. In this work, a boxblur algorithm was applied. But it would work just as well with a Gaussian blur. This is open to the individual preferences of how the influence should spread.

0.000	0.000	0.000	0.000	0.000
0.000	**1.000**	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.000
0.000	0.000	**-1.000**	0.000	0.000

Table 3.1: Boxblur influence example grid - Iteration 0

The Box Blur is a spatial domain linear filter. This takes a pixel (or in our case a cell) from the grid and takes itself and its surrounding pixels and calculates the average as the new value. A 3 by 3 box blur (radius 1) can also be described here as a matrix:

$$K = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{3.6}$$

K is the average value of pixel and its surrounding values.[Handwiki, 2021]

0.250	0.200	0.200	0.030	0.008
0.240	**1.000**	0.160	0.006	0.007
0.210	0.180	0.150	0.040	0.002
0.060	-0.040	-0.070	-0.100	-0.010
0.005	-0.170	**-1.000**	-0.200	-0.078

Table 3.2: Boxblur influence example grid - Iteration 1

0.420	0.370	0.290	0.080	0.025
0.400	**1.000**	0.250	0.009	0.027
0.300	0.250	0.140	0.030	-0.007
0.068	-0.057	-0.130	-0.15	-0.069
-0.039	-0.221	**-1.000**	-0.271	-0.142

Table 3.3: Boxblur influence example grid - Iteration 2

Through this box blur, the influence is gradually expanded with 1,000 and -1,000 and then stagnates after a certain iteration. The Celle with 1.000 means that there is an allied unit that places its influence there and -1.000 means that there is an enemy unit.

3.3 Waypoints

Waypoints are positions distributed around the game world. With waypoints, the AI can use this for its pathfinder in order to progress in the game world. Tactical waypoints require more data describing these points in order to make a correct decision about which waypoint to use [Mellington, 2020].

Chapter 4

Experimental setup

The experiment is a match against two sides in the style of "conquest" as in games of Battlefield. In Conquest, there are a certain number of places that a side tries to capture. At the beginning, each place is still uncaptured. When a team has taken a point, it always gets points added to their points account at certain time intervals. The team that has reached a certain number of points first after a certain time has won. One side will consist of a squad of five agents, the other side of five squads of one agent each. A squad always knows where its agents are and whether a unit sees a friendly or enemy agent and can thus build up an IM. Each time a team wins, this is saved in a file and the next match starts from the beginning. This allows you to run this several thousand times so that the result is not falsified.

4.1 Assumption

I think the side with the squad and its five agents has a slightly higher win rate than the side with the 5 squads with one agent each. This is because it has more information to decide, for example, which of the points to attack first or which point may have no enemy units.

4.2 Rules of the game

The goal of one side is to reach 100 points to win the game. This means that there will always be 3 points distributed on the map, each of which can be captured. A point is captured when an agent is in a certain area without any other enemy agents. Each point adds one point to the score every 5 seconds. Each agent has the option to shoot and kill an enemy agent. Each agent has 100 lives and can shoot a projectile with 10 damage every 3 seconds. If the projectile hits a wall or another enemy agent it is destroyed. If an agent has 0 life it is destroyed and after 10 seconds it is re-instantiated at a random location on the map but not in a capturable point and thus rejoins the game.

4.2.1 Playing field construction

4.2.2 Randomness factor

Chapter 5

Results

Chapter 6

Discussion

Chapter 7

Conclusion

Chapter 8

Lists

List of Acronyms

IM influence map

FOW fog-of-war

AI artificial intelligence

GDC game developers conference

List of Figures

List of Tables

Bibliography

Daniel Brewer and Rez" Graham. Knowledge is power: An overview of knowledge representation in game ai, 07 2020. URL https://www.youtube.com/watch?v=Z6oZnDIgio4&t=994s.

Alex J. Champandard. The core mechanics of influence mapping, April 2021. URL https://www.gamedev.net/tutorials/programming/artificial-intelligence/the-core-mechanics-of-influence-mapping-r2799/.

Handwiki. Box blur, December 2021. URL https://handwiki.org/wiki/Box_blur.

Ian Mellington. *AI for Games*. Third edition edition, December 2020.