Terrain Analysis and Artificial Intelligence: Gaining a Positional Advantage

Authors:
Nicholas Colotouros
Terrence Ko

<u>1 Introduction:</u>

The main objectives of the project were to analyze terrain in games using various metrics and then have an artificial intelligence use it to easily gain terrain advantage.

We analyzed the performance of a simple AI on different methods of terrain analyses against naive bots and cheating bots. The methods used are heat maps and the metrics for them are based on the height, shooting distance, and view distance. The implementation was done in Unity 5 and scripts were used interchangeably depending on the situation (ie.naive or AI). Our hypothesis was that heat maps could allow a developer to create a simple AI that would have interesting behaviours.

After running tests, we can determine that height-based heat maps created the more realistic behaviours for AI bots, highlighting the importance of high grounds or covers in maps.

<u>Implementation separation:</u>
Nicholas: Level creation, Terrain analysis, game logic and back end
Terrence: Level design, artificial intelligence and related classes

<u>Report separation:</u>
Nicholas: Introduction, methodology overview, all terrain analysis parts, conclusion
Terrence: Introduction, background, all artificial intelligence parts, conclusion

<u>2 Background</u>

Creating a challenging AI that is not too computationally expensive and competent is one of the more common challenges that developers have to face when it comes to developing RTS and FPS/TPS genre games. We have chosen to analyze the performance of an AI that is computationally light yet is sufficiently competent that it can emulate decisions that would be taken by a human player. This approach is most apt for RTS genres where decisions for units are relatively straightforward: destroy the enemy or attempt to gain the upperhand before engaging. When it comes to having the upperhand, it does not suffice to have more units; the terrain control is equally if not more important. Terrain advantage is usually defined as control of the high ground, much like in real life. In FPS games, high grounds allow to add an extra dimension to attack vectors and in RTS games, they allow for better unit spacing.

The use of heat maps is not limited to RTS games and can be used in virtually any game where units can move freely. In fact, the use of heat maps is quite well documented in Killzone (Stratman et al., 2005): said maps are used to create procedural combat tactics.

The heat maps in Killzone allowed the AI to plan routes based on the safety of the destination as well as the risk needed during the trip.

Using information on heat maps, map designers can create competitive game levels that would pace each match appropriately. Path accessibility have a correlation with heat. For examples, chokepoints have low accessibility and very high heat. Such information can be precomputed and be fed easily into the AI (Ontañón et al.) in order to reduce computation during games.

The end result that we seek is to create simple GOAP AI that will provide us with a simple yet satisfying challenge.

3 Methodology

The game created is a simplified version of StarCraft. There are two teams, red and blue which are given a number of units to control. Each unit has the same speed, firing radius, damage output and firing rate.

Similar to StarCraft, it is advantageous to have the higher ground. In our implementation, units with the higher ground cannot be shot by units on the lower ground, but the units with the lower ground can be fired upon by the units on the higher ground if they are in the firing radius.

A unit having higher ground is a full platform above the unit on the lower ground. A unit on the ramp connecting platform x to platform y where y is higher than x can be fired upon by units on both platforms x and y. However, the moment it gets to platform y the units on platform can no longer fire upon it.

Three levels were created to test terrain analysis:
Level01 (bowl): A flat level with a lower ground in the center accessible from both sides.
Level02 (stage): A flat level with a high ground platform only accessible from the left side.
Level03 (canyon): A level consisting of a canyon/choke point where the higher ground is accessible from the north and south sides.

Each level was analyzed using 3 different metrics: height, view distance and shooting distance. Then a red team and blue team was placed on top. Each unit exerting it's own influence on the terrain, changing the heat. As the units move, so does their influence, which decays the further it is from the unit and goes out roughly with the same radius as the unit's shooting radius.

3.1 Terrain Analysis:

One thing to note about each form of analysis is that once values are determined they are scaled such that it fits within the maximum terrain heat specified by the user.

3.1.1 Height

The simplest and most naive method. The heat of each tile is determined by it's height. Low ground is considered to be the most dangerous and therefore the lowest tiles have the highest heat and high ground is considered to be more favourable which means lower heat values.

The logic behind this is that high ground is always favourable to lower ground because it means your opponent will never be able to shoot you without you being able to shoot back.

Once the height has been determined, 0 is used as a normal by convention of the map building. Heat is then determined by the following algorithm where max/min Height are the y coordinates highest and lowest tiles and maxTerrainHeat is specified by the user what the highest terrain values are:

```
extreme = Max(maxHeight, abs(minHeight))
heatIncreaseRate = maxTerrainHeat / extreme

for each tile t in the board
    t.heat = heatIncreaseRate * t.position.ycoordinate * -1
```

The multiple of -1 ensures that high heat is reserved for terrain below a height of 0.

3.1.2 View Distance

The second method used to analyze the terrain was view distance. Heat for a given tile is determined by the number of tiles that can be seen by a unit standing on it. This assumed infinite distance with no obstructions.

The way this is done is that a dummy unit is spawned, moved onto the first tile and then raycasts of infinite length are shot from the top of each tile to the dummy unit. The number of raycasts that hit the dummy unit is the number of tiles that can be seen from that spot. The value is recorded and the process is repeated for each tile on the map.

As a result of the raycasts coming from the top of the tiles, the dummy unit will never get a successful raycast from a tile that is higher ground and is not a ramp. Ramps on the same height will still succeed.

Once this is complete, the heat must be normalized such that the spot that can see the most tiles gets the lowest possible heat rating, the spot that can see the fewest tiles has the highest heat rating possible and everything in between scales linearly. The algorithm for calculating the heat is:

```
heatIncreaseRate = 0;
visionDiff = maxVision - minVision
if( visionDiff != 0)
    heatIncreaseRate = 2 * maxTerrainHeat / visionDiff

**for each** tile t in the board
    t.heat = maxTerrainHeat - heatIncreaseRate * (t.tilesSeen - minVision)
```

Max/min vision are the highest and lowest recorded number successful raycasts for a given spot. maxTerrainHeat is specified by the user what the highest terrain values are. tilesSeen are the number of successful raycasts recorded for when the dummy unit was on that tile.

The reason there is a multiple of 2 in the heatIncrease calculation is that values go from -maxTerrainHeat to +maxTerrainHeat and the intent is to use all values in that range.

3.1.3 Shooting Distance

This process is identical to the View Distance method except the raycasts have the same length as the firing radius of the unit. As a result only tiles that can be shot at will score recorded raycasts.

3.2 Artificial Intelligence

3.2.1 Overview

3 simple AI models were developed for this research: one uses the heat maps and scans the area to determine safe locations to proceed in. The two others are naive AIs. The first one merely advances forward and attacks anything on site. The last one has “cheating” abilities and will immediately seek out and attack an opposing bot without having to scan the terrain first.

A navmesh was used to implement movement AI and bots fire immediately when the enemy is sufficiently close. Mouvement suspension is not required for attacks. Units on the lower ground may not attack units on the upper ground, fact that will dominate the decision of the AI agents.

Using precomputed values from heat maps, we can construct a rather efficient behaviour tree for the AI. The main goal of the AI is to achieve territorial dominance (cross to the other side of the map) or complete extermination of the opponent. In the interest of self preservation, AI units only engage if there is a clear numeric advantage. The heat map greatly simplifies non-trivial behaviour such as group mouvement and threat assessment.

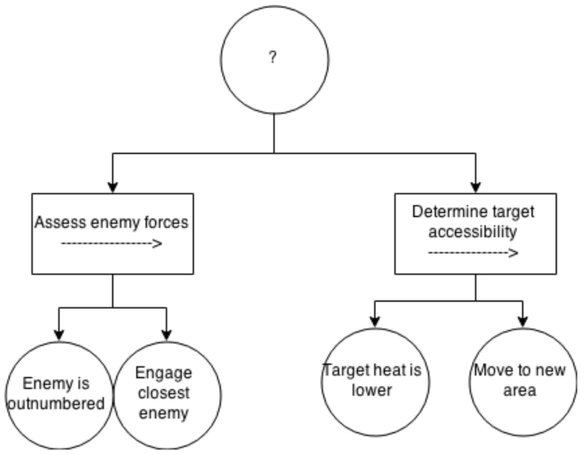


Figure 1. Behaviour tree of our experimental AI.
The use of heat map lightens the complexity of our AI.

A squad manager was implemented in order to keep track of the number of blue and red units. This information is also used by the cheating AI to pinpoint the closest opponent in order to attack it.

4 Results:

4.1 Terrain Analysis

Terrain analysis was done with a maximum heat value of 2, so it can drop all the way to -2 and a shooting radius of 3.

Note: Unity consistently renders the ramp colours darker than their values actually are due to the inclination and lighting. We did our best to minimize this but ultimately it was impossible to eliminate completely. Please rely on the numbers when looking at the ramps..

4.1.1 Height

See the first images of figures 4.1, 4.2 and 4.3 in the appendix for terrain analysis using height.

Results were as expected with all 3 levels. Any tile with a height above 0 in the y axis got a lower heat value, tiles with exactly 0 height for a heat value of 0 and tiles below 0 on the y axis got higher heat values.

4.1.2 View Distance

See the second images of figures 4.1, 4.2 and 4.3 in the appendix for terrain analysis using view distance.

Something of note in levels 1 (basic) and 3 (canyon) is that the ramp is considered the best view point. I believe that a possible explanation for this is that when the tile in question is higher ground and there is lower ground nearby due to the angle and that the raycast comes from the center of the tile it is not necessarily successful. This is a limitation on the accuracy of the method used.

The tiles also don't get a bonus for being on higher ground. So while moving to the tile at the top of the ramp won't quite see as many tiles as the ramp, the unit standing on it will be impervious to enemy fire from the lower ground. However, this metric does not account for unit health or firing range.

4.1.2.1 Level 1: Bowl

A very interesting anomaly is how the plain area away from the basin has a lower heat value than the edges of the basin on the higher ground. You would think that the limitation of accuracy discussed earlier would make the edge of basin would be able to see more.

4.1.2.2 Level 2: Stage

Unlike the other two maps the ramp is not as favourable in heat as the plateau in the center. I believe this is due to the larger nature of the level so the plateau obstructs a number of tiles from successfully raycasting to the dummy unit on the ramp ramp as the angle will be so small that the ray will not make it over the blocks, but if you're on top of those other blocks then you can see those tiles.

Something of note in the context of this level is that while being near the plateau generates the least favourable heat values in the level, those areas would theoretically be the best areas for taking cover. While this doesn't apply in this game, this kind of approach might have been useful for a stealth game as it minimizes how many people can see you.

4.1.2.3 Level 3: Canyon

In addition to what was mentioned at the beginning of 4.1.2, the choke point was identified as the most dangerous spot, as it has the lowest area of sight.

The outer corners don't have as favourable of a heat as the inner ones where the chokepoint is because the level ends and there is nothing to overlook on those sides.

4.1.3 Shooting Distance

See the third images of figures 4.1, 4.2 and 4.3 in the appendix for terrain analysis using shooting distance.

One thing that came up across all 3 maps was that the corners were given high heat values. This is because while the rest may have been open space, the level ended so there were not tiles to see on 3 sides. For this kind of game the corners themselves would be rather unfavourable spots.

Like with view distance, the ramps score as very favourable positions. I believe that while view distance suffered to a degree from lack of accuracy, in this case I think the larger influence is from the lack of giving a bonus to tiles for having true high ground.

4.1.3.1 Level 1: Bowl

The most interesting part of this level is the lower ground. The center is almost as favourable as the edges of the higher ground. This is because aside from the ramps vision from the center is almost completely free of obstructions for a vision radius of 3.

4.1.3.2 Level 2: Stage

One thing that was interesting was that the center of the high ground is the least favourable spot on the entire map.While tactically this is not true, being in that spot means only being able to shoot one tile on the lower ground, which is the one on the right side.

Another thing of interest is that the larger, flat, open areas on the side of the map are the most favourable parts rather than the high ground, which is very much unlike the View Distance analysis. This is because those areas can see all tiles covered by the field of view while the corners of the high ground cannot see the lower ground from the opposite side of the platform. For example if a unit is in the north west corner of the platform then a raycast from the low ground south of the position will fail due to the other corner of the platform being in the way.

4.1.3.3 Level 3: Canyon

The result for this map is actually very similar to the results from the view distance method. The choke point has nearly the same heat. The only major difference is that the left and right sides of the map are less favourable than being one or two tiles towards the center (going along the x-axis) because of the map ending.

4.1.4 Comparisons and Discussion of terrain analysis:

Height analysis goes for the global maximum of terrain advantage when sometimes a local minimum is the most favourable spot. A possible improvement might have been to consider heat based on height relative to tiles within a certain radius.

One of the major drawbacks to view distance and shooting distance analysis was that there was no consideration whatsoever for having the high ground. As a result the ramps tended to score more favourably. An improvement to these methods would have been to either combine them with the height analysis method some how or give a bonus for being on tiles with higher ground.

In terms of performance, height analysis is O(n) and view/height distance analysis is O(n^2) where n is the number of tiles on the map. Even with our small maps, calculating view distance and shooting distance are quite costly to the point where you can see a noticeable startup time. In terms of practicality it's not of any consequence when there are static maps as you can just save the heat map after calculating it once and then load it without calculating it on subsequent times for that map.

Overall it seems that view distance and shooting distance are better than height, each with their own advantages and disadvantages in different levels.

4.2 Artificial Intelligence

Bots are able to emanate their own heat field and this is used by the behaviour tree to automatically cluster together allied (blue) units. Behaviours were somewhat erratic at times due to unreliable raycasting. However, the results of the use of heat maps for AI design are quite convincing.

It is to be noted that when there are enough bots together, the higher heat from lower grounds can be overridden and the group proceeds to engage the enemy even if its position may be disadvantageous. Such behaviour is not uncommon amongst human players. Heat generated from height values generally offered better performances to heat-using AIs and as such, we will focus results using this metric. Results from view distance and shooting distance based heat maps can be found in the appendix.

Important: the main objective for the teams is to reach the other side. If both teams succeed and have the same number of bots, there will be no engagement as this would result in random results (first bots to shoot win). As such, draw results have been discarded from the result tables.

4.2.1 Level 1: Bowl

Naturally, the heat inside the bowl discourages the AI bots from directly passing through the middle. However, whenever there were enough blue units to form a cluster, the blue units would still engage the red units through this passage, often obtaining mixed results. The blue units would however actively avoid the "bowl" if they had lost an ally and would proceed through the sides to attack the red unit or reach the other side.Naive and cheating AIs did not exhibit peculiar behaviour in their attack patterns

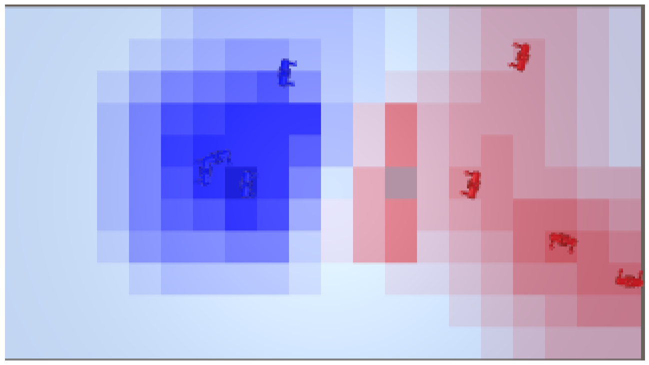


Figure 2. Blues are about to enter the bowl

Blue | Red	Using heat	Naive	Cheating
Using heat	--	Red wins 70%	Red wins 80%
Naive	Red wins 90%	--	Red wins 50%
Cheating	Blue wins 50%	Blue wins 60%	--

Table 1. Win rates over 20 games in bowl scenario(4v4)

<u>4.2.2 Level 2: Stage</u>

By design, this level vastly favours the blue team and in a series of unofficial testing, the blue team would win 100%. The high ground was always at least occupied by one blue unit, offering fire support to the blue units that were on the ground. Additionally, because a blue unit was present on the high ground, proximate lower grounds were deemed safe enough for blue units. Consequently, even if red units could destroy a few blue units on the ground, they would always lose territorial dominance to the blue units, either retreating to the corners to be safe, or be destroyed altogether. Naive AIs were easily destroyed upon engagement and cheating red AIs were often unable to overcome the territorial advantage of the blue units.

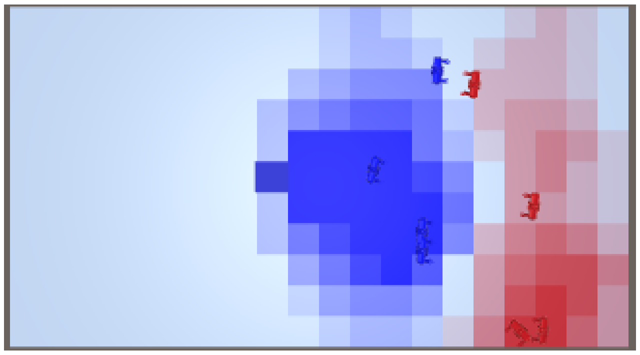


Figure 3. A blue unit has taken position on the high ground, giving fire support to its allies.

Blue | Red	Using heat	Naive	Cheating
Using heat	Blue wins 90%	Blue wins 85%	Red wins 65%
Naive	Red wins 55%	--	Red wins 70%
Cheating	Blue wins 60%	Blue wins 70%	--

Table 2. Win rates over 20 games in stage scenario(4v4)

4.2.3 Level 3: Canyon

This level implements the classic chokepoint that is very important in RTS games such as Starcraft. When two heat-using AI bots are used, the game ends in a stalemate 60% of the time. This was expected as the ramps are very narrow, making uphill access impossible without suffering catastrophic damage. Naive bots would attempt to access the other side through the easiest route, that is, the canyon. Bots using the high ground were able to eliminate opposing naive bots without any losses 80% of the time. Cheating bots were able to gain the upperhand most of the time due to the fact that they occupied the high ground much faster.

Blue | Red	Using heat	Naive	Cheating
Using heat	--	Blue wins 100%	Red wins 90%
Naive	Red wins 90%	--	Red wins 80%
Cheating	Blue wins 65%	Blue wins 80%	--

Table 3. Win rates over 20 games in canyon scenario (4v4)

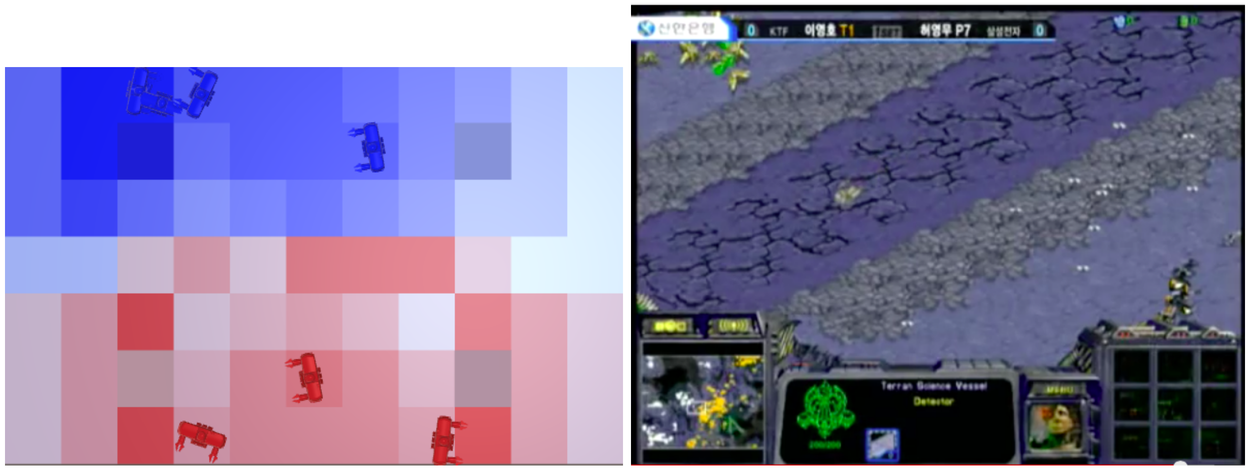


Figure 4/Image 3. A example of stalemate in the canyon and a similar situation in a Starcraft pro game. Note the minimap in the game and the canyon separating the forces.

<u>5 Conclusions/Future work:</u>

The different metrics surprisingly gave varying results when it comes to AI performance. It would seem that although the AIs could engage in combat behaviour in a consistent manner, a slightly more complex behaviour tree would have been beneficial to the AI bots in the event that terrain information is insufficient (such as when al bots were on the same ground). Unit coordination would be the next logical step in our AI implementation as such a thing requires a lot of threat analysis, which is simplified by heat maps.

The project could be expanded into using influence maps and terrain analysis to assist with squad movement. The vision based terrain analysis can be expanded into the realm of stealth game level design by taking guard movement into consideration. Different unit types can project heat in different manners, increasing the complexity of AI behaviours. When it comes to territories in RTS games, it could be possible to implement and adaptation of heat maps to create zones of interest (such as resource rich places) to drive units to these areas. By using heat maps, game management computation can be directed to action planning (such as GOAP driven decisions) and create light but competent AI systems.

6 BIBLIOGRAPHY

Perkins, L. 2010. Terrain analysis in real-time strategy games: An integrated approach to choke point detection and region decomposition.
https://www.aaai.org/ocs/index.php/AIIDE/AIIDE10/paper/viewFile/2114/2563

Santiago Ontañón, Gabriel Synnaeve, Alberto Uriarte, Florian Richoux, David Churchill, Mike Preuss. A Survey of Real-Time Strategy Game AI:Research and Competition in StarCraft, http://webdocs.cs.ualberta.ca/~cdavid/pdf/starcraft_survey.pdf.

Straatman, R. & van der Sterren, W. & Beij, A (2005). Killzone's AI: dynamic procedural combat tactics, http://www.cgf-ai.com/docs/straatman_remco_killzone_ai.pdf

7 Appendix

View distance heat map on scenarios 1,2,3 in order in 4v4 maps:

Blue | Red	Using heat	Naive	Cheating
Using heat	--	Red wins 85%	Red wins 90%
Naive	Red wins 90%	--	Red wins 80%
Cheating	Red wins 65%	Blue wins 65%	--

Blue | Red	Using heat	Naive	Cheating
Using heat	Blue wins 85%	Blue wins 55%	Red wins 90%
Naive	Red wins 85%	--	Red wins 80%
Cheating	Blue wins 50%	Blue wins 75%	--

Blue | Red	Using heat	Naive	Cheating
Using heat	--	Blue wins 65%	Red wins 80%
Naive	Red wins 85%	--	Red wins 70%
Cheating	Blue wins 65%	Blue wins 75%	--

Mirror matchups were discarded unless there is an unbalanced map.

Shooting distance heat map on scenarios 1,2,3 in order in 4v4 maps:

Blue | Red	Using heat	Naive	Cheating
Using heat	--	Blue wins 50%	Blue wins 60%
Naive	Red wins 90%	--	Red wins 70%
Cheating	Blue wins 65%	Blue wins 80%	--

Blue | Red	Using heat	Naive	Cheating
Using heat	Blue wins 85%	Blue wins 95%	Red wins 90%
Naive	Red wins 50%	--	Red wins 80%
Cheating	Blue wins 70%	Blue wins 80%	--

Blue | Red	Using heat	Naive	Cheating
Using heat	--	Blue wins 55%	Blue wins 55%
Naive	Red wins 90%	--	Red wins 60%
Cheating	Blue wins 60%	Blue wins 80%	--

Mirror matchups were discarded unless there is an unbalanced map.

Figure 4.1: From top to bottom: Height, View Distance and Shooting Distance for the bowl level

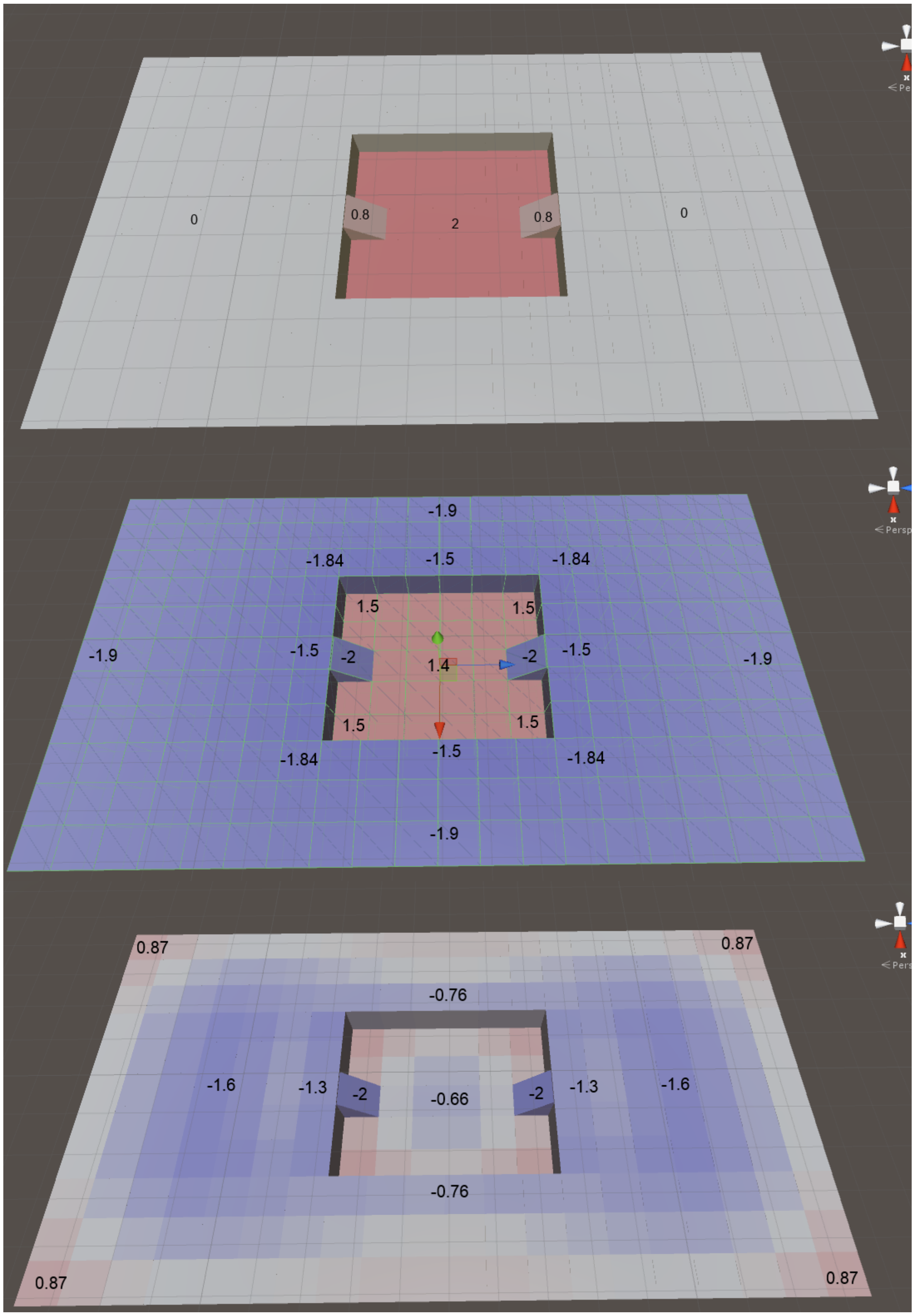


Figure 4.2: Top to bottom: Height, View Distance and Shooting Distance for the stage level

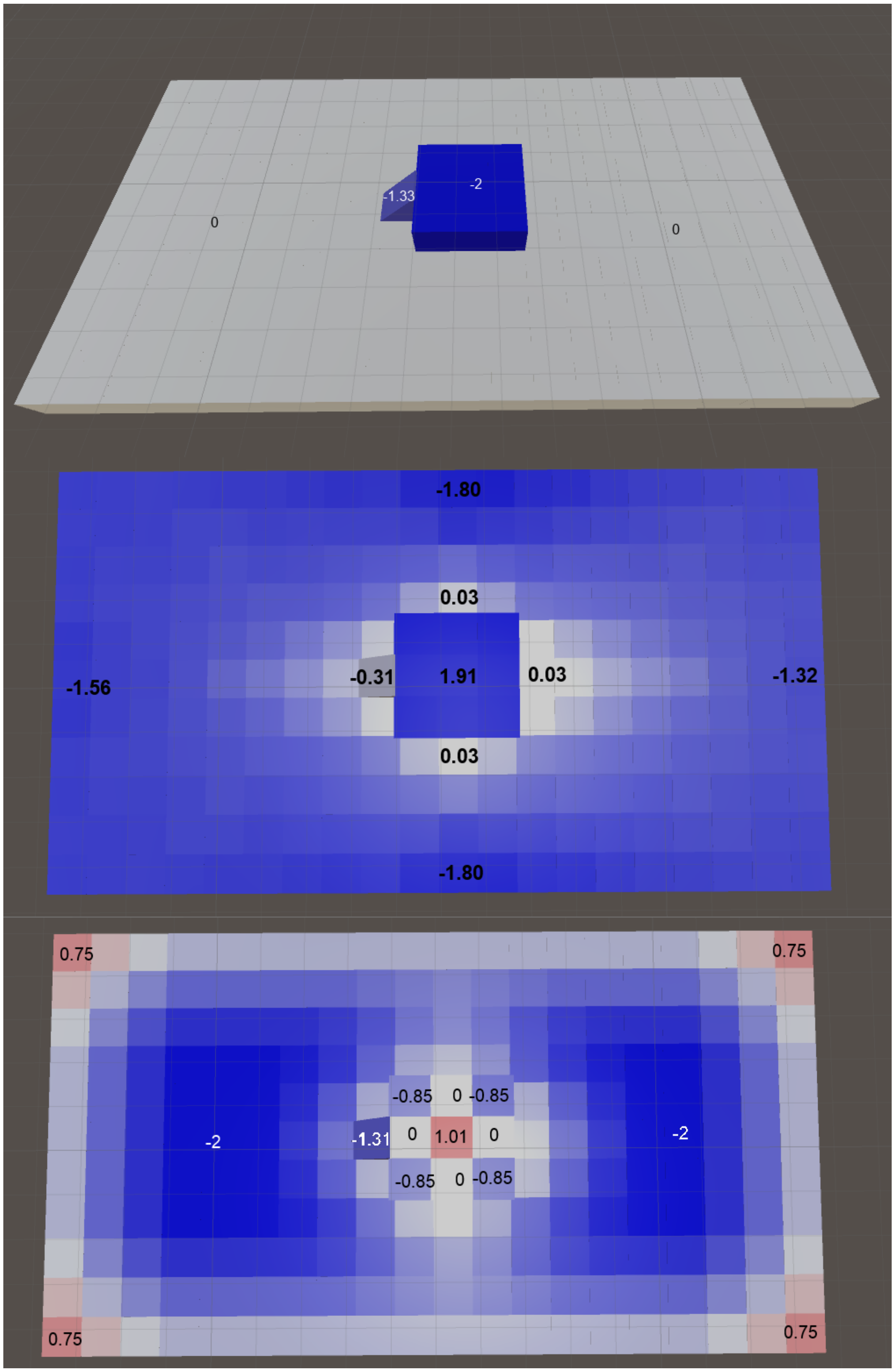
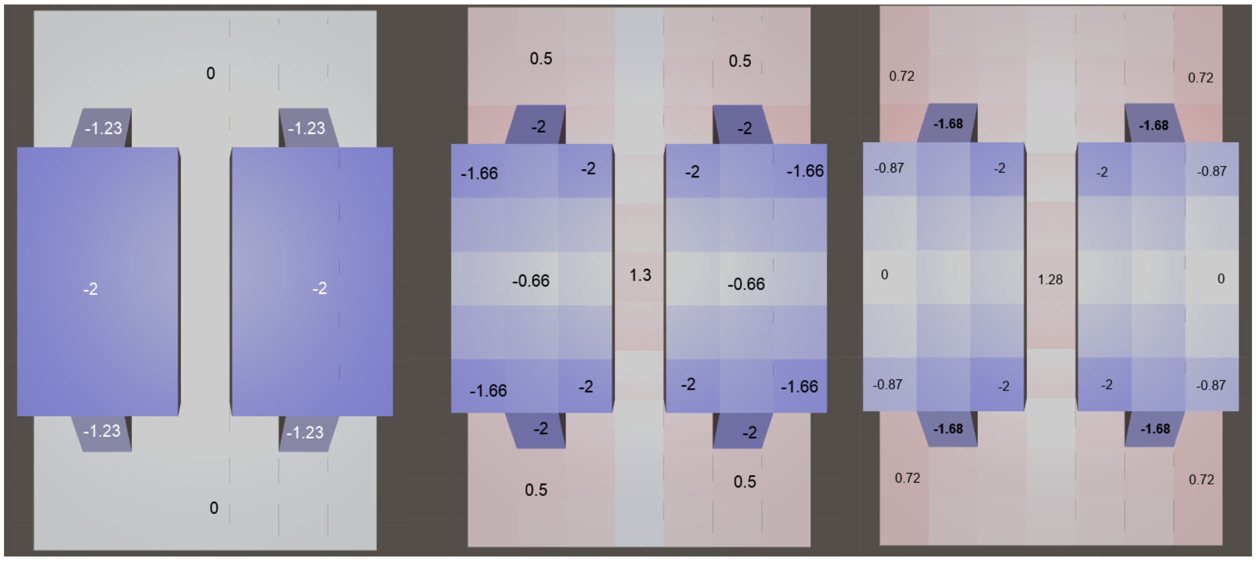


Figure 4.3: From left to right: Height, View Distance and Shooting Distance for the canyon level


0
-1.23
-1.23
-2
-2
-1.23
-1.23
0
0.5
0.5
-2
-2
-1.66
-2
-2
-1.66
-0.66
1.3
-0.66
-1.66
-2
-2
-1.66
-2
-2
0.5
0.5
0.72
0.72
-1.68
-1.68
-0.87
-2
-2
-0.87
0
1.28
0
-0.87
-2
-2
-0.87
-1.68
-1.68
0.72
0.72