4

What You See Is Not What You Get

Player Perception of AI Opponents

Baylor Wetzel and Kyle Anderson

4.1 Introduction

For many, even in the game development community, the goal of AI development is to make the "winningest" AI that one can from the perspective of the developer, which sounds like a reasonable goal. From the perspective of the person spending money to buy the game, of course, an unbeatable AI is not such a great idea. The player just wants to have fun.

If we want to make an AI that the player appreciates, we need to understand how the player sees the AI. So we decided to do exactly that. A group of players played a turn-based strategy game against two dozen AI opponents; ranked them on difficulty, realism, and fun; and explained what they believed each AI's strategy was. Among the findings, players are bad at determining AI's strategies and detecting randomness and cheating. Perceived difficulty correlated highly with realism, but actual difficulty's correlation was only moderate. Complex techniques often achieved higher scores but seemed no more difficult, realistic, or fun to players than significantly simpler AIs. No factor correlated highly with how much players enjoyed playing a particular opponent.

4.2 The Game

We have made AI opponents, NPCs, and systems for a wide variety of genres, and we know that different genres have different AI requirements. Our goal was to learn something universal, but our test had to be grounded in a specific game. We wanted something where the AI controlled an opponent since it allowed the player to test the AI more. We wanted something where the AI lasted more than a few seconds (this turns out to be shockingly rare). We wanted the AI to have a lot of choices since it gave the player more opportunity to study the AI. We wanted symmetric game play and equal resources, by which we mean that, just like in chess and very much unlike most boss battles, we wanted both the human and the AI to have the same options and chances of winning. We reasoned that if the player faces the same decisions as the AI, then the player more easily understands the AI's reasoning. In perhaps our most controversial decision, we decided we wanted a game where time was not a factor; our belief being that a player facing a clock has less time to worry about what the AI is thinking.

In the end, we decided to build a turn-based strategy game. We based our game on the combat portion of *Heroes of Might & Magic*, a well-balanced game giving players and the AI many strategic options.

In our game, each player controls 10 armies, where an army is a stack of units of the same type. The player picks among 46 types of units such as elf, devil, hydra, crusader, ice dragon, and bone dragon. Unit properties included obvious candidates such as health, damage range, attack, and defense (damage multiplier and divisor, respectively) as well as speed (controls which unit goes first), damage types (fire, poison, etc.), their resistances, and, finally, cost. Each player gets the same amount of money to purchase their armies. In our test, however, we required both sides to use random armies each game, again in the belief that it would force the player to think more about their—and the AI's—strategy.

As in *Heroes*, there are damage-over-time attacks, some units that are strongly biased to offense or defense and some units that are significantly stronger than others. It is not uncommon to see 500 imps fight one black dragon. Who the AI decides to attack is a clue to the AI's goals and thought process, and we want to know if the player can figure out what the AI is thinking. As area of effect attacks, favored by bosses in boss battles across the world, show little of an AI's intelligence, we removed them from the game.

For those familiar with *Heroes of Might & Magic*, we made a few other changes to simplify the game; removing morale modifiers and adding and tweaking a few units to better support balance and strategy use. The most significant change is that we removed movement and thus any need for spatial reasoning—units fought from where they stood and could attack any other unit. This allowed players to have more choices of who to attack, again allowing players to have more opportunities to examine the AI's reasoning. We tried to give the player every opportunity to see the AI's decision-making in action.

As in *Heroes*, aside from damage range, there is no randomness in the game. Units do not miss. This places the focus on decision-making (important for this study) and deemphasizes the effects of luck.

To summarize our game, both sides are given a random set of units. The AI has to decide which of it its armies to employ and which enemy armies to attack. The armies vary significantly in size, strength, and defense. They potentially employ elemental attacks and resistances as well as damage-over-time effects. The player cannot see the AI's thought process, but they do see the AI make dozens of decisions each game and have ample time to see and think about those decisions.

4.3 AI Design Goals

Before we can conduct experiments on what the player does and does not notice and appreciate in an AI opponent, we need to decide what is important to us as AI designers in developing an AI. We decided on a few things. For the player, we care about the AI's overall competence, visible error rate, visible error severity, and the player's perception of the AI. For the developer, we care about the AI's complexity and implementation difficulty. The AIs we created for the experiment were designed with these goals in mind.

For us, player perception meant the player's opinion of the AI's difficulty and realism and how much fun it is to play against. As part of this, we created AIs that exhibited certain common human behaviors (persistence and revenge, explained in the next section). We also created AIs that attempted to subtly allow the player to win. One thing we did not include was AIs with deep and somewhat irrational personalities. As is discussed in the results section, this did not prevent players from believing we had. A scientific investigation of how personality characteristics affect a player's perception of an AI agent is an interesting topic for a future experiment.

Complexity refers to how complicated an AI's programming is. Although not commonly used in commercial games, it is common for programmers to consider using techniques such as deep learning, game theory, Markov chain Monte-Carlo simulations, and genetic algorithms. Implementation difficulty refers to how hard it is to construct an AI. A complex AI does not necessarily take a long time (especially if one uses a library). A simple AI that uses a lot of variables might take little time to create but a significant amount of time to tune. For both factors, the relevant question is, "do players notice, appreciate, or care about the AI technique or complexity?" Put another way, games are not cheap to make, are we spending our time and money on the right things?

By overall competence, we mean how often does it win? Visible error rate refers to mistakes the AI makes that the player notices. By visible error severity, we do not mean that the AI makes a mistake that causes it to lose the game, we mean that the AI makes a mistake that causes the player to think, "Wow, this AI is really stupid." If an AI opponent beats a player but makes an incredibly stupid mistake that no human would make, would the player be impressed or would they focus on the incredibly dumb error? We first seriously considered this issue when a billiards AI which one of the authors played against made a difficult shot that only a very good player could make; then missed three simple shots, which any amateur could make. The AI made the wrong types of errors, forcefully reminding the player that he or she was just playing a computer and causing him or her to never play the game again. Those sorts of mistakes are perhaps more obvious in a billiards game than a turn-based strategy game, but it is an issue that game AI developers in all genres should think about it.

4.4 Visible AI Traits

What can the player tell about their AI opponent? We certainly do not expect players to notice or care whether the AI was implemented using a behavior tree, influence map, or some form of deferred, time-sliced AI technique. But what do we expect them to notice? This important design question influences the types of AIs we want to make. For this experiment, we decided on eight properties of an AI opponent: use of available information, use of direct versus summary information, technique complexity, predictability, strategy obviousness, persistency, revenge, and hustling.

4.4.1 Available Information

When we talk about whether an AI uses the available information, we are talking about how many relevant and available pieces of information the AI chooses to use. A strategy such as *Attack the Enemy with the Highest Attack Score* only requires one piece of information. *Attack the Enemy You Can Do the Most Damage To* requires at least two—the amount of damage you can do and the defensive ability of the enemy. *Attack the Enemy You Can Kill This Turn* requires at least three—the damage you can do, the enemy's defensive ability, and the enemy's current health. If the targeting strategy accounts for damage type (e.g., using fire attacks against wooden enemies or avoiding using poison attacks on metal enemies), two more pieces of information are required. If every game AI used all available, relevant information, it would not be worth measuring in this experiment, but we have played with enough triple-A game AIs that fought fire with fire (or fire-resistant creatures with fire attacks) to know better.

4.4.2 Direct versus Summary Information

Assuming it knows how, an AI can calculate how powerful a given enemy is by using low-level information about the unit. For example, unit strength might be determined by its attack, defense, and starting health. These values are easy to compare across units. Information such as which unit attacks first is harder to weigh relative to health and attack strength. Special abilities (depending on the ability) can be hard to evaluate. For example, if a unit in a first person shooter has a 10% chance of not consuming ammunition when attacking, does that make them stronger or weaker than a similar unit with 5% more health? The AI designer could create a complicated formula that accounts for all of the relevant factors and then balances the weights of each term, or they could take the fast and easy approach of using something that presumably summarizes all that information such as the unit's cost. Want to attack the most dangerous enemy? Attack the one that costs the most. Not all games have unit cost, but some have level and in others maximum health is a decent proxy for overall unit strength. This is what we mean by use of direct versus summary information. Using summary information is easier and potentially captures the values of hard to weigh factors such as special abilities that would otherwise take a lot of time (and therefore money) to properly weigh. It might (or might not) result in a less capable AI but the important question is, will the player notice?

4.4.3 Technique Complexity

Technique complexity refers to how complicated an AI technique is. A complex technique might be complicated because it uses a complex formula, or it might be complex because it uses a fancy AI technique. A simple technique, in contrast, is probably simple because it only took a few minutes to think up and implement. The latter has a variety of advantages ranging from taking less time to implement (and therefore less chance of missing a deadline) and less cost (because it takes less time to implement) to being easier to debug. The downside is that the AI might be dumber and dumber, AI opponents might be less fun to play against. This assumes that the player can tell the difference between a fancy AI and a simple one, which was one of the questions in this experiment.

4.4.4 Predictability

Predictability refers to how predictable an AI's actions are. Strategy obviousness refers to how easy it is to explain what the AI is doing. If an AI picks its targets at random, the AI's actions are essentially impossible to predict, but the AI's target selection strategy is obvious. This experiment checks that supposition.

4.4.5 Persistency

Persistency refers to the AI's tendency to stick with a choice. An AI that selects an enemy and never let them go is persistent. An AI that switches targets every turn is not. We ran this experiment multiple times over the course of a year. Earlier versions of this test showed that humans tended to be persistent, and an AI that switched targets too frequently was seen as "computery." We added this trait after those early tests.

4.4.6 Revenge

Another human-like trait was revenge. Human players who were attacked by a given enemy tended to focus on that enemy, punishing it for attacking them. Adding this behavior to an AI is quick and easy. In theory, it gives the AI more personality and makes it appear more human.

4.4.7 Hustling

We use the term hustling instead of cheating because the intention behind the two is different. When we say someone is cheating, we often mean they are trying to get a win they did not earn to immediately benefit themselves. When we say a person is hustling you, we often mean the person is pretending to be worse than they are, actively managing the game so that when you lose, it is only by a little and you feel like you almost had it. When we use the term hustling in game AI, we are talking about an AI opponent who makes small, undetectable but plausible changes that result in the player just barely winning. We have two goals: the player must believe the AI is trying its best and feel accomplished when they beat the AI. We do not want the AI to make visible errors and, if the AI is so far ahead that there is no way for the player to win, we want the AI to ruthlessly crush the player so that when they finally do beat the AI, they feel like they beat a worthy opponent. For some of our AIs, hustling was done by shaving dice rolls when the AI was significantly ahead (e.g., if it normally does 20–40 points of damage, lower the range to 20–30). Others also picked the second best target if that target was also reasonable (within 20% of the value of the top ranked target). One AI switched strategies to a second, still good, strategy if it was too far ahead. It is OK (and often computationally efficient, which is important when you have limited computational resources) to cheat, but it is critical that the AI not get caught cheating. If done correctly, the player should not notice that the AI is cheating but should feel like they won all on their own.

4.5 The AIs We Created

We tried to create AI opponents who varied both across the aforementioned AI properties and within them (e.g., multiple low-complexity techniques). In the end, we created 51 AI opponents, although only 22 were used in these experiments.

The most complicated AIs were *PowerCalculationFancy*, which used a complicated, manually tuned formula, and *TeamDamageMonteCarlo*, which used simulations to calculate how much collective damage the enemy team could do before and after a given target was attacked. Less complicated AIs included *HighestStackHealth*, which picks the enemy army whose combined health is the highest, and *HighestStackAdjustedHealth*, which picks the enemy army whose combined health times their defense is the highest. Low-complexity AIs included *Alphabetical* and *LongestName*, which chose enemies based on their names. These were obviously not written to be good AIs, but they are good tests of whether players can figure out what an AI is thinking. They also provide a floor on AI performance, making it easier to evaluate more serious AI approaches.

HighestStackCost uses unit cost as a proxy for power, attacking the stack whose units cost the most. *PersistentHighestStackHealth* and *PersistentHighestStackCost* pick a target by health and cost, respectively, then stay on that target until they are destroyed. *IndividualRevenge* attacks the first unit that attacks them and stays on them until they are defeated. In *TeamRevenge*, every AI attacks the first enemy to attack any of them.

RandomBot attacks random enemies. *TK* attacks random teammates (it is not a very good AI). *TopDown* attacks the army closest to the top of the screen. We created these AIs as a baseline of how well players could infer an AI's strategies. *Wolf's* strategy was to pick off the weak and damaged, which was implemented by attacking at random until a unit falls below 50%, at which point every AI army attacks that unit. *Gentleman* only picks on enemies its size (where size = level; if there are none, it picks on bigger enemies and, as a last resort, smaller ones). *AlternatingStrategies* switches between the strategies of *HighestStackCost* and *HighestUnitHealth*. *PingPong* assigns each enemy a score and chooses one of the top ones, the one chosen being random but weighted by their score.

When *ShareBear* has an army that has damage-over-time effect, it looks for enemy units not already affected by it (damage-over-time damage does not stack in the game). If there are none, it attacks the stack that does the most damage.

HighestStackAdjustedHealthManaged, *HighestStackCostManaged*, *PowerCalculationFancy*, and *TeamDamageMonteCarloManaged* are hustling AIs. They determine how well they are doing relative to the player and nerf their die rolls when too far ahead. They never improve their chances since players are more likely to notice (and complain) when an AI does better than normal than when they do worse. *TeamDamageMonteCarloManaged* also switches strategies to the still pretty good *HighestStackAdjustedHealth* if it is too far ahead.

4.6 The Test

This experiment was repeated four times, but the results reported here come from the final iteration, which was conducted with 12 game design students. It is not a huge sample size but given the qualitative nature of some of the data, the consistency with earlier results, and the resources we had available, it is not a bad one.

Subjects were asked to play multiple games against each of 22 AI opponents, tracking their wins. All games were played on the same day, and the order of opponents was randomized. After each set of games, they rated the AI opponent on fun, realism, and difficulty on a five-point scale. They also wrote a short explanation of what they felt the AI's strategy was.

4.7 The Numbers

We will start with how well the AIs did. Table 4.1 shows how many games each AI won. The top scoring AI was *ShareBear*, whose distinguishing feature is that if it has an army capable of doing damage-over-time (DoT) attacks, it targets enemy armies not currently suffering from a DoT effect. Otherwise, it attacks the army that does the most collective damage (but does not take into account the army's attack score, making the metric misleading). The next highest scoring AI used a hand-tuned formula that took into account a number of variables. Doing roughly as well was a one-move look-ahead Monte-Carlo algorithm. The worst performing AIs used nonsense strategies such as attacking armies based on their name, order on the screen, or simply selecting ones at random.

It is unclear what to make of the performance of the hustling algorithms. *PowerCalculationFancy* won 71% of its games, whereas *PowerCalculationFancyManaged*

Table 4.1 Actual Difficulty of AI Opponents

Rank	AI	Wins
1	ShareBear	76%
2	PowerCalculationFancy	71%
3	TeamDamageMonteCarlo	70%
4	PowerCalculationFancyManaged	57%
5	HighestStackCostManaged	54%
6	HighestStackCost	52%
	PingPong	52%
7	IndividualRevenge	38%
8	HighestStackHealth	37%
	PersistentHighestStackCost	37%
9	AlternatingStrategies	33%
	HighestStackAdjustedHealthManaged	33%
10	PersistentHighestStackHealth	30%
	TeamDamageMonteCarloManaged	30%
11	HighestStackAdjustedHealth	28%
12	Alphabetical	26%
13	LongestName	23%
	Randombot	23%
14	Wolf	22%
15	Topdown	13%
16	TK	9%
17	TeamRevenge	7%

Table 4.2 Perceived Difficulty of AI Opponents

			Actual Difficulty	
Rank	AI	Score	Wins	Rank
	HighestStackAdjustedHealthManaged	3.6	33%	9
2	HighestStackAdjustedHealth	3.5	28%	11
	IndividualRevenge	3.5	38%	7
	PersistentHighestStackCost	3.5	37%	8
	PersistentHighestStackHealth	3.5	30%	10
	RandomBot	3.5	23%	13
	TeamDamageMonteCarlo	3.5	70%	3
	TeamDamageMonteCarloManaged	3.5	30%	10
3	TeamRevenge	3.4	7%	17
4	HighestStackCost	3.3	52%	6
	Topdown	3.3	13%	15
5	LongestName	3.2	23%	13
	PingPong	3.2	52%	6
	PowerCalculationFancy	3.2	71%	2
	Wolf	3.2	22%	14
6	AlternatingStrategies	3.1	33%	9
	HighestStackCostManaged	3.1	54%	5
	HighestStackHealth	3.1	37%	8
7	TK	3	9%	16
	Alphabetical	3	26%	12
	ShareBear	3	71%	1
8	PowerCalculationFancyManaged	2.8	54%	4

won 57%. Given that the goal of a hustling AI is to put up enough of a fight to give a competent player a challenge without constantly winning, a 57% win rate is close to what we were hoping for. In every instance, the hustling version of an algorithm won fewer games than the nonhustling version, but in the case of the Monte-Carlo algorithm, it dramatically underperformed, 30% versus 70%.

The scores for perceived difficulty (Table 4.2) are far less spread out, with all but one of the AIs scoring between 3.0 and 3.6 on a five-point scale and more than a third scoring between 3.5 and 3.6. The hustling AIs were seen by players to be just as difficult as the nonhustling version, even though the actual scores show otherwise, implying that the players did not notice that the AIs were cheating for the player's benefit. Interestingly, the top scoring AI was perceived as the second easiest AI to beat while the second hardest AI was perceived to be the same difficulty as the AI that attacked units based on how long their names were.

Table 4.3 shows how realistic the players felt each AI was, and Table 4.4 shows how much they enjoyed playing each one. Of interest is that *LongestName*, a strategy that is utterly absurd, which ranked fourth for realism and fifth for fun. *Alphabetical* ranked sixth and seventh, respectively. We think we can explain why and do so in the next section.

Table 4.3 Perceived Realism of AI Opponent

Rank	AI	Score
1	PersistentHighestStackCost	3.8
	TeamDamageMonteCarlo	3.8
2	HighestStackHealth	3.7
	PowerCalculationFancy	3.7
	ShareBear	3.7
3	HighestStackCostManaged	3.6
4	LongestName	3.5
	PowerCalculationFancyManaged	3.5
5	TeamRevenge	3.4
6	Alphabetical	3.3
	HighestStackAdjustedHealthManaged	3.3
	HighestStackCost	3.3
	TopDown	3.3
7	AlternatingStrategies	3.2
	PersistentHighestStackHealth	3.2
	PingPong	3.2
	RandomBot	3.2
	TeamDamageMonteCarloManaged	3.2
8	HighestStackAdjustedHealth	3.1
	IndividualRevenge	3.1
9	wolf	3.0
10	TK	1.2

The earlier tables contained a few surprises but not as many as Table 4.5. Table 4.5 shows how many subjects were able to determine each AI's strategy. Full points were given if they described something roughly similar to one of the main points of the AI's strategies and partial credit if they described a relevant, if not core, aspect of the strategy. We were rather lenient on grading but even so, and despite all the opportunities the players had to study the AI opponents, strategy recognition was nearly nonexistent. The only two AIs whose strategies were discovered by the majority of the players were *TK*, which attacked its own teammates, and *TopDown*, which attacked the army closest to the top of the screen. The first has a unique, memorable, and shocking behavior, whereas the second leverages the player's spatial reasoning ability (which this experiment seems to show plays a major role in reasoning even in domains where it should not be relevant). 83% of players were unable to recognize an AI that was literally nothing more than a random number generator.

It is our opinion that this is an important finding—players have no idea what your AI is doing. We did not expect players to recognize a Monte-Carlo algorithm in action. We feel it is important but unsurprising that players could not differentiate between fancy AI (*PowerCalculationFancy*), simple AI (*HighestStackHealth*), and AI using indirect measures (*HighestStackCost*). We hoped players could not tell the difference between AIs that were trying their hardest and those that were intentionally throwing the game to make the player feel more competent. But we were surprised that players were completely unable to

Table 4.4 Enjoyment of AI Opponent

Rank	AI	Score
1	HighestStackAdjustedHealthManaged	3.6
2	HighestStackAdjustedHealth	3.5
	IndividualRevenge	3.5
	PersistentHighestStackCost	3.5
	PersistentHighestStackHealth	3.5
	RandomBot	3.5
	TeamDamageMonteCarlo	3.5
	TeamDamageMonteCarloManaged	3.5
3	TeamRevenge	3.4
4	HighestStackCost	3.3
	TopDown	3.3
5	LongestName	3.2
	PingPong	3.2
	PowerCalculationFancy	3.2
	Wolf	3.2
6	AlternatingStrategies	3.1
	HighestStackCostManaged	3.1
	HighestStackHealth	3.1
7	Alphabetical	3.0
	ShareBear	3.0
	TK	3.0
8	PowerCalculationFancyManaged	2.8

detect even blatant and intentionally human-like behaviors such as having an entire team focus on the first person to attack them (*TeamRevenge*), nor could they tell when an AI was changing strategies (*AlternatingStrategies*).

(An aside: While the average player might be unable to quickly determine an AI's strategy and is likely uninterested in conducting experiments to puzzle it out, The Internet is a large place and if your game has enough players, eventually someone will figure it out and publish it somewhere. We leave it to you to decide whether that is a bad thing or, perhaps, an opportunity to add Easter eggs for your more highly motivated players.)

Table 4.6 shows how the various measurements are correlated. Correlation scores are on a scale of 1 to –1, with 1 meaning they move together perfectly, –1 meaning they move in opposite directions, and 0 meaning there is no connection between them. Note that strategy recognition correlations are omitted because players were unable to recognize all but two strategies, making the analysis meaningless.

Perceived difficulty strongly correlates with everything but fun. Actual difficulty strongly correlates with perceived difficulty and moderately correlates with realism. In both cases, difficulty correlated with realism. This is perhaps in keeping with anecdotal findings from others that making an enemy more difficult made the AI appear more intelligent. Fun, unfortunately, did not correlate with anything. How much one enjoys playing against an AI opponent appears to be independent of how difficult or realistic the AI is.

Table 4.5 Player Recognition of AI Strategies

Rank	AI	Correct	Partial	Score
1	TK	12	0	1.00
2	TopDown	9	1	0.79
3	RandomBot	2	0	0.17
4	ShareBear	1	2	0.17
5	PersistentHighestStackCost		4	0.17
6	HighestStackAdjustedHealth		1	0.04
	PersistentHighestStackHealth		1	0.04
	TeamDamageMonteCarloManaged		1	0.04
7	Alphabetical			0.00
	AlternatingStrategies			0.00
	HighestStackAdjustedHealthManaged			0.00
	HighestStackCost			0.00
	HighestStackCostManaged			0.00
	HighestStackHealth			0.00
	IndividualRevenge			0.00
	LongestName			0.00
	PingPong			0.00
	PowerCalculationFancy			0.00
	PowerCalculationFancyManaged			0.00
	TeamDamageMonteCarlo			0.00
	TeamTevenge			0.00
	Wolf			0.00

Table 4.6 Correlation between Measurements

	Perceived Difficulty	Actual Difficulty	Realism	Fun
Perceived Difficulty	1.00	0.80	0.81	0.04
Actual Difficulty	0.80	1.00	0.54	−0.19
Realism	0.81	0.54	1.00	0.18
Fun	0.04	−0.19	0.18	1.00

4.8 The Explanations

Players determined an AI's strategy by studying which of the player's armies they attacked each turn. We were interested in two things. First, did they have enough information to determine properties of the AI such as its strategy and the complexity of that strategy? Second, what did they think the strategies were?

To understand the player's reasoning, it helps to have an example. Assume that the player has the armies shown in Table 4.7. They play four opponents who, on the first turn, attack their armies in the order specified in Table 4.8.

Maximum damage is the base damage done by one unit. Adjusted damage is computed as the unit damage times the number of units times the attack score. Inflicted damage in combat is adjusted damage divided by the target's defense.

Table 4.7 An Example Set of Armies Owned by a Player

# Units	Army	Unit Cost	Health	Maximum Damage	Attack	Defense	Speed
4	Black Dragon	8,000	400	160	50	60	7
6	Ice Dragon	4,000	300	70	50	50	4
1	Dwarf	33	12	3	11	11	3
1,650	Imp	20	7	2	10	10	6
80	Minotaur	200	70	10	16	16	6
12	Monk	500	50	20	30	22	5

Table 4.8 The Order in which a Player's Armies were Attacked by the Opponent

Opponent A	Opponent B	Opponent C	Opponent D	Opponent E
Black Dragon	Imp	Imp	Monk	Black Dragon
Ice Dragon	Black Dragon	Minotaur	Ice Dragon	Ice Dragon
Monk	Ice Dragon	Black Dragon	Minotaur	Minotaur
Minotaur	Minotaur	Ice Dragon	Black Dragon	Dwarf
Dwarf	Monk	Monk	Imp	Monk
Imp	Dwarf	Dwarf	Dwarf	Imp

Since we are only looking at the first turn of combat, we cannot determine some of the AI properties described in the previous section. We omit revenge, persistency, guided randomness, and hustling, but there is still information that both we and the players can glean and valuable points to make. (A multiarmy, multiturn example illustrates poorly in book form.)

Take a few seconds to see if you can figure out what the strategy of opponent A is. In doing so, you are placing yourself in the mind of the player and experiencing the game the way the way they are (just as you have Table 4.7, the players had a spreadsheet of all unit values). Is there an obvious explanation for why the AI chose the armies they did? The answer is probably "yes." In fact, there are probably several, a point we will come back to in a minute. Many of the players wrote "They are selecting the strongest," which is true, depending on how you define strongest. In this case, Opponent A picks the armies whose units have the highest maximum damage score.

Opponent B is less obvious. Take a second to think about what the AI might be thinking. You might notice that the AI no longer picks the dragons first, it picks the imps, which are the weakest units in the game. Why? If you look over the army details, you would likely notice just how many imps are there. While an individual imp only does two points of damage, the stack of imps does a cumulative 3,300 points of damage, five times what the black dragons do (4 * 160 = 640).

Opponent C's attack order is slightly different than Opponent B's. As mentioned above, the actual damage an army does is the unit damage (Opponent A) multiplied by the number of units to get the stack damage (Opponent B) multiplied by the unit's attack score to get the adjusted damage. Opponent C uses this formula. Just because this strategy is reasonable, simple, and straight forward does not mean players will figure it out. Nor should we expect them to. Although more complex than Opponent B's strategy, it produces almost exactly the same results. It also presents the exact same results as the AIs *Stack Cost*, *Adjusted Stack Health*, *PowerCalculationFancy*, and potentially others (e.g., *TeamDamageMonteCarlo*). Moreover, in fact, no player was able to determine the strategy of any of these AIs.

What are the lessons here? One might be that players are simply not perceptive, deep thinking, or creative, even in environments where they are told to study the AI (we will come back to this theory in a bit). Another might be that reasonable and perhaps semireasonable strategies tend to come to the same conclusions. A corollary is that complicated strategies using multiple variables that require a lot of tuning might not produce results that the player will notice or appreciate. Another lesson might be that, in many games, traits are correlated. Powerful units such as dragons tend to have high health, attack, and damage, whereas weak units such as dwarves are relatively low on all of these. A corollary is that the AI might not have much information to exploit unless game designers intentionally put in units with unusual sets of values. We did that in this experiment by making units that had extremely high attack relative to health and defense and vice versa for the sole purpose of giving players and the AI more strategy options; an intelligent AI is wasted on a game whose design does not allow the AI to be clever.

The strategy used by Opponent D is not readily apparent from this snapshot. The attack order here is random. There is no other strategy—the AI is literally a random number generator. After several rounds of switching targets with no apparent reason, it is obvious that this AI has no strategy. Or so we thought. 83% of players were unable to tell the AI was acting randomly. This was especially surprising as other AIs with consistent strategies were routinely accused of being random. *RandomBot* was actually rated one of the least random AI opponents.

Before discussing Opponent E, let us discuss some of the strategy guesses that we received from players. Although far from exhaustive, several representative ones are listed in Table 4.9 (Note: The table does not include answers of the form “I do not know,” which was common for almost all AIs).

A few observations:

1. Few players correctly guessed any portion of what AI was doing.
2. Different players often had very different impressions of the AIs (e.g., *HighestStackHealth* was said to “go for tough guys” by one player and “shoot the group with the most guys” by another; “groups with the most guys” tend to have weak units because one cannot afford large numbers of tough units).
3. More than half the AIs were accused of having no strategy and just behaving randomly.
4. One of the AIs that did not cheat was accused of blatant cheating. One of the AIs that cheated in the player’s favor was accused of blatant cheating against the player.
5. Nonsense AIs are believed to be acting intelligently.
6. Many of the AIs were said to intentionally spread poison throughout the team in order to weaken them over the long run. This was often considered to be the AIs’ only strategy. Despite this, only one of the 22 AIs knew how to use DoT attacks such as poison, and none of the AIs used them if they did not have units that could perform them (units were chosen at random, and only two of the 42 units had them).
7. Described strategies sometimes ascribed personality and prejudices to the AI.

Table 4.9 Examples of AI Strategies as Determined by Players (Spelling Left Unchanged)

AI	Strategy	Comments
Alphabetical	Select unit by name	• Takes out units by base strength • Attack the advatar with the highest attack • Spread damage around ot lower my dps • Take out the strongest one by one
HighestStackCost	#units * cost	• Attacks the stacks with the highest attack • Attack the advatar with highest def • Poison and take out the strongest
HighestStackCost Managed	#units * cost Lowers its damage dice rolls when player is significantly behind	• Randomly attacking groups • Attacks highest stacked with his front monsters and loweest stacked with there back monsters • It mimics the first two attacks from the opposing team and then generates a random attack pattern
HighestStackHealth	#units * health	• Shoot the group with the most guys • Go for tough guys • This AI likes punching babies
PersistentHighestStack Health	#units * health Stick with enemy until it dies	• Killing the lowest costing creatures first • This AI attacks the group with the highest amount of units • Attack strongest first • Takes out dot first then strong mid then strongest guy
HighestStackAdjusted Health	#units * health * defense	• Appears to attack randomly • Randomly choose 1 of the top 3 highest total life • Attacks the units with the most speed • It kills everything in one or two hits regardless of the stats
IndividualRevenge	Attack first enemy that attacks it	• Random • Wittle down the big hitters • DoT uses poison units to attack my weakest units. Chaos would attack power units • Goes after middle strongest and spreads the damage • Attacks based on what type of unit it is, only attacks all Ice Wolves etc. Then moves to the next unit
LongestName	Attack enemy with longest name	• Attacks the stack with the most health • Attacking the highest dps • Focus fire • Strongest, then strongest DoT, then Strongest big group
RandomBot	Select enemies at random	• Attack the advatar with highest def • Spreads damage among all enemies' attacks strongest and works way down spreading damage • This AI attacks two lowest staked creatures than two highest stack creatures

(*Continued*)

Table 4.9 (*Continued*) Examples of AI Strategies as Determined by Players (Spelling Left Unchanged)

AI	Strategy	Comments
TeamDamage MonteCarlo	Use one-turn look ahead with Monte-Carlo Algorithm	• This AI takes very randomly • Appears to be a cheating AI that attacks too many times per turn • Attacked all ranged first and then attacked at random • Take out the highest attack the slowest
TeamDamage MonteCarloManaged	Use one-turn look ahead with Monte-Carlo Algorithm. If winning, switch to adjusted stack health strategy	• I do not have a clue, the AI must be pulling a random creature out of its hat to attack • Random flailing • Cheating AI seems to attack too many time per turn • hates medusa's bandits and orcs, do not even mention hydras • This AI took me out behind the wood shed!
TeamRevenge	All armies focus on first enemy to attack to attack any of them	• Appears to attack randomly • Attack the highest costing creature, then the third lowest, repeat step one then two • It attacks the celestial characters like the angels and the monks first

It is the last point we find the most interesting. Perhaps we are bad game designers, but it did not occur to us to give the AIs personality, which, in retrospect, seems at odds with our goals—players, being human, are designed to be good at inferring motivations, intentions, prejudices, and personality quirks—not mathematical formulas and optimizations. Many of our players saw personality in our AIs where none existed. One player believed that *HighestStackCostManaged* was copying their moves. *TeamRevenge* was said to prefer attacking holy characters such as angels and monks. *TeamDamageMonteCarloManaged* apparently hates medusas, bandits, orcs, and hydras. In the *Heroes* games, every unit belongs to a type (life, order, nature, etc.), and these are all chaos monsters. Chaos monsters are apparently treated specially by *IndividualRevenge*, who was said to use their chaos armies to attack the player's most powerful units while the rest spread poison among their army.

It is not clear why players believed these things, which is bad because it means it is out of the game designer's control. In some cases, however, the reason for the player's perception makes sense. *IndividualRevenge* was said to have to be out for ice wolves. In truth, ice wolves are the fastest unit in the game and thus attacks first. Since *IndividualRevenge* attacks the first unit that attacks it, it makes sense that it would attack ice wolves when they are present. The behavior is not personal, simply an artifact of strategy and speed scores, but the player does not see revenge, they see an AI with an unreasonable hatred of their beloved dogs.

This brings us back to Opponent E. Take a look at the order of armies the AI attacked. The AI starts with black dragons and ice dragons before working its way down to monks and imps. When asked what strategy the AI was following, the player explained to me that the AI hated dragons. They then went on to explain a very vivid backstory for the AI that involved their family being killed by dragons when they were young and their subsequent quest to rid the world of them. The actual AI strategy was to pick targets based on how long their names were. Dragon, however, is not a short word, and fantasy convention says dragon names should have modifiers—black dragon, green dragon, ice dragon, bone dragon, chaos dragon, and so on. As a result, all dragons had names longer than all other units and *LongestName* therefore did indeed have a bias for dragons. This personality might not have been apparent had we not had dragons in the game (the order minotaur, dwarf, and monk lacks the cohesive story that black dragon, bone dragon, and ice dragon have). Sometimes character is an accidental byproduct of other, seemingly unimportant game design decisions.

4.9 Conclusion

Through this experiment, we learned a few things about players and a few things about ourselves. We will not argue that all of these things are true for all games, and we would love to hear from other developers about what works for their games. We believe there are a lot of interesting lessons here that apply to a wide variety of games, and savvy game designers will know which ones are likely to be helpful to their games.

Players are not particularly good at estimating the difficulty of an AI opponent. They get the general order OK (but not great; the AI that won the most was considered one of the easiest to beat) but tend to think all but the worst AI are of roughly equal difficulty whereas the win/loss records suggest quite the opposite. Being a turn-based strategy, one would assume the player had ample opportunity to study the AI's ability. This is less likely to be true in fast paced in-games; although in many other game types, enemies are not more difficult because they are more intelligent, they are more difficult because they have more health, weapons, and so on. So perhaps we should say that players are not particularly good at telling how intelligent their opponent is.

Is it worth investing time in complicated AIs? The news is mixed. The most complicated AIs performed better than simple AIs, and the simple AIs outperformed the truly ridiculous AIs. However, players did not seem to notice. An AI that chose actions at random was rated by players as one of the most difficult AI opponents, tying with the Monte-Carlo algorithm and placing significantly ahead of the top two scoring AIs.

Perceived difficulty correlated highly with realism—if players thought an AI was difficult (or, as we said earlier, more intelligent), they also thought it was more realistic. Unfortunately, the correlation between actual difficulty and realism was not as high. As a result, it is hard to know how to interpret this. Do players feel like a given AI is more realistic because they believe, erroneously, that it is more intelligent, or do they believe it is more difficult and intelligent than it is because it seemed realistic?

We added two human-like traits to AI opponents: revenge and persistency. We also had an AI (Wolf) that changed tactics when a unit was seriously wounded, all units pouncing on the weakened enemy to finish it off. Each of these behaviors came from human studies on player personality. None of the AIs with these human-like traits scored highly on realism. The AI considered the most realistic was a Monte-Carlo-based algorithm that was arguably the least human-like AI in the test. We are not sure what this says about how players perceive realism. Perhaps the player's inability to understand what the AI was thinking left them little else to base realism on than how hard (they thought) it was to beat.

As perhaps comes as no surprise to anyone, players are not very good at detecting cheating. No player commented on the several AI opponents who routinely cheated in their favor, but several were quite vocal about how much the AI cheated to beat them, even though no such AI existed.

How good are players at inferring an AI opponent's strategy? Horrible, it turns out. We did not expect them to know whether an AI was basing its decision on cost or health, but they showed very little insights into any aspect of the AI's reasoning, even when the AI followed simple, consistent rules. We looked for patterns in qualitative measures such as "this AI is more random than others," "this AI uses a more sophisticated technique," "this AI considers several factors," and "this AI is as dumb as a rock" but did not find them. Between the player's inability to accurately gauge an opponent's difficulty/intelligence or understand the AI's strategy, we found no evidence that the players we tested appreciate the amount of effort AI designers put into making an AI opponent, at least not the effort put into making the AI intelligent.

Although we did not attempt to measure it, it is our belief (and the belief of many others over the decades) that players appreciate effort put into AI if it is the right kind of effort. It is common to hear that players loved an AI's external signs of intelligence, meaning the AI said out loud what it was thinking. Many players might not recognize or appreciate an AI that flanks them (the AI behaving intelligently), but they do appreciate when an AI explicitly shouts "let's flank them" (the AI claiming it is behaving intelligently). In game AI, words sometimes speak louder than actions. Our experience in other games with emotional characters suggests players react strongly to NPCs that cry, laugh, shiver, and otherwise outwardly show signs of thinking. We suspect that if we had included an AI that said before each attack "attack the wolves, their speed is a threat to us" or "we can finish off that giant, everyone focus on him," it would have been perceived as significantly more intelligent than the others. We think players appreciate intelligence; the tests here simply indicate that they do not seem to be very good at recognizing it. Perhaps the lesson to take away is that it is the AI designer's job to make sure they do not miss it.

What makes a game fun, at least when it comes to an AI opponent? We do not know. In this experiment, how much the player enjoyed playing a particular AI opponent did not correlate with actual or perceived difficulty or intelligence, nor did it correlate with how realistic the opponent was considered. It also did not correlate with the player's ability to recognize, understand, or appreciate the AI's strategy because the players could not. Hustling AIs that tried to give the player a chance to win without letting them know are placed in the top two spots for fun AI, but hustlers are also placed in two of the three bottom spots so it clearly is not that. It truly surprised us that a random AI tied for second as the most fun AI. Our initial thought was that the game we used must be poorly designed if a random AI is seen as just as viable as purposeful ones, but we reminded ourselves that our test bed is based on one of the most popular games in history.

This brings us to our final observation—players invent stories for the nonplayer-controlled characters in the game. They see cheating, bias, motivations, and desires where none exist. And while we did not capture it in our spreadsheets, anecdotally it seemed that players enjoyed the game more when they felt their opponent had that level of personality. If we wanted the player to have more fun, perhaps we should worry less about optimizing the AI for winning and worry more about giving it the appearance of desires, motivations, biases, and back stories.