Measuring player experience on runtime dynamic difficulty scaling in an RTS game

Johan Hagelbäck and Stefan J. Johansson *Member IEEE*

Abstract—Do players find it more enjoyable to win, than to play even matches? We have made a study of what a number of players expressed after playing against computer opponents of different kinds in an RTS game. There were two static computer opponents, one that was easily beaten, and one that was hard to beat, and three dynamic ones that adapted their strength to that of the player. One of these three latter ones intentionally drops its performance in the end of the game to make it easy for the player to win. Our results indicate that the players found it more enjoyable to play an even game against an opponent that adapts to the performance of the player, than playing against an opponent with static difficulty. The results also show that when the computer player that dropped its performance to let the player win was the least enjoyable opponent of them all.

I. INTRODUCTION

The important thing is not winning but taking part!?
The saying origins from the Olympic Creed that once was formulated by the founder of the Olympic Committee, Pierre de Coubertin, in the beginning of the last century [1]:

> The important thing in the Olympic Games is not winning but taking part. Just as in life, the aim is not to conquer but to struggle well.

These words are, as we will argue, indeed applicable also to computer games.

We have made a study on player experience of five different computer opponents in an RTS game, two with static difficulty setting and three which changes difficulty setting dynamically during a game. Traditionally difficulty settings of computer opponents in games are set manually and as the players learns to play the game, they reach the points where the opponents are no longer a challenge for them. Another issue can be that the challenge step between the pre-defined difficulty levels is too wide. A player might find a difficulty level too easy, but raising the level one step is a too big challenge for him. A third issue is that a player might discover a weakness in the computer opponent's tactics, and once discovered the player wins easily by exploiting the weakness.

The study was carried out during DreamHack Winter 2008, the largest LAN party in the world. The event is held yearly in Jönköping in Sweden and attracted more than 13.800 participants [2]. The participants in our experiments played a game against one of the five bots in an RTS game and were asked to fill in a questionnaire after the game has ended. A total of 60 persons participated in the study.

Both authors are at the School of Computing, Blekinge Institute of Technology, Ronneby, Sweden (phone: +46-457-385831; email: {jhg,sja}@bth.se).

A. Real Time Strategy Games

In real-time strategy, RTS, games the player has to construct and control an army of soldiers and vehicles and use it to destroy the opponent forces. Typically the player has to gather resources to construct buildings which in turn allows the player to build attacking and defensive units. The game runs in real-time in contrast to turn-based strategy games such as Civilization. Famous titles in the genre is Command & Conquer, Warcraft, Starcraft and Age of Empires.

B. Measuring Enjoyment in Games

There are several different models of player enjoyment in computer games, ranging from the work of Malone in the early 80's on intrinsic qualitative factors for engaging game play [3], [4], to the work of e.g. Sweetster and Wyeth on the Gameflow model [5]. We will in our paper not try to model the enjoyment as such, e.g by partition it into factors. Instead we let the players express their experience in terms of a number of adjectives of six different clusters relating to the strength, the variation and the enjoyment and their opposites (weakness, predictability, and boredom). We then analyse the players' opinions about the enjoyment to their opinions about the strength and the variation of the computer opponent. This does not measure the enjoyment in any absolute way (and that is not our intention either), but relate it to properties of strength and variation.

C. Dynamic difficulty scaling

Difficulty scaling means that the difficulty of a game is adjusted to suit the skills of the human player. The purpose is to give the player a challenge even when his skill in the game increases. It is typically set manually by the player by choosing from different pre-set difficulty levels (e.g. Beginner, Normal, Hard, Expert). Another approach is to use built-in adaption of the difficulty in the game AI. Bakkes et al. describe an approach called *rapidly adaptive game AI* where the difficulty in the RTS game Spring is adapted at runtime by using observations of current game state to estimate the probable outcome of the game [6].

Olesen et al. describe an approach where several factors that contribute to the difficulty were identified, and artificial neural network controlled agents that excel on those factors were trained offline and used in dynamic difficulty scaling [7].

Both approaches use an evaluation function to estimate the current strength of the player relative to that of the AI.

D. Outline

We will first go through the environment which we will use to conduct the experiments, followed by a short description of the bot that we use and the modifications made to it. In Section III we will describe the experimental setup and the results are presented in Section IV. We finish by discussing the results and the methodology, drawing conclusions and line out directions for future work.

II. THE OPEN REAL TIME STRATEGY PLATFORM

Open Real Time Strategy (ORTS) [8] is an open source real-time strategy game engine developed by the University of Alberta as a tool and test-bed for real-time AI research. Through an extensive scripting system the game engine supports many different types of games. In the experiments performed in this paper we used the Tankbattle scenario. In Tankbattle each player start with five bases and 50 tanks each. The bases are randomly placed on a 1024x1024 field with 10 tanks positioned around each of them. In the field there are also cliffs that the tanks cannot pass, and sheep that move around and that could not be run over. The goal is to destroy all the bases of the opponent. The number of units and bases is fixed and no additional units can be constructed during the game.

A. Multi-Agent Potential Fields

The Multi-Agent Potential Field based bot used in the experiments is based on previous work we conducted in [9], [10]. The idea is to generate potential fields by placing attracting or repelling affectors at interesting positions in the game world, for example enemy units and buildings. The different fields are weighted and summed together to form a total potential field which is used by the agents, tanks, for navigation.

We identified four tasks in the Tankbattle scenario: Avoid colliding with moving objects, Hunt down the enemy's forces, Avoid colliding with cliffs, and Defend the bases. Three major types of potential fields are used to handle the tasks: *Field of Navigation*, *Strategic Field*, and *Tactical field*.

The field of navigation is generated by letting every static terrain tile generate a small repelling force. We would like our agents to avoid getting too close to objects where they may get stuck, but instead smoothly pass around them.

The strategic field is an attracting field. It makes agents go towards the opponents and place themselves at appropriate distances from where they can fight the enemies. The field is created out of subfields generated by all enemy tanks and bases. The generated subfields are symmetric with the highest, i.e. most attractive, potentials in a circle located at Maximum Shooting Distance (MSD) from the enemy unit or structure (see Figure 1. The reason for placing the most attractive potential at MSD is that we want our tanks to surround and fight the enemy at the maximum distance of their cannons instead of engaging the enemy in close combat. This is illustrated in Figure 2. It shows an own tank attacking an enemy unit from maximum shooting distance.
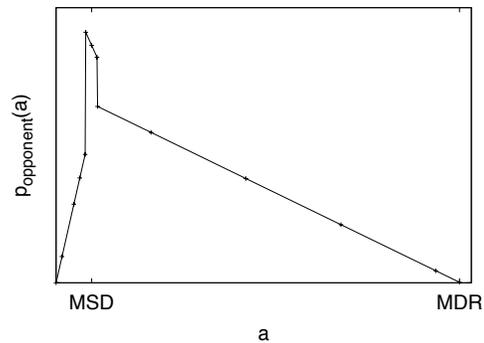

Fig. 1. The strength of the attracting force generated by an opponent. MDR is the *Maximum Detection Range*, i.e. the distance from which the opponent starts to attract a unit. MSD is short for *Maximum Shooting Distance*, the range of the weapons of, in this case, both sides' tanks.
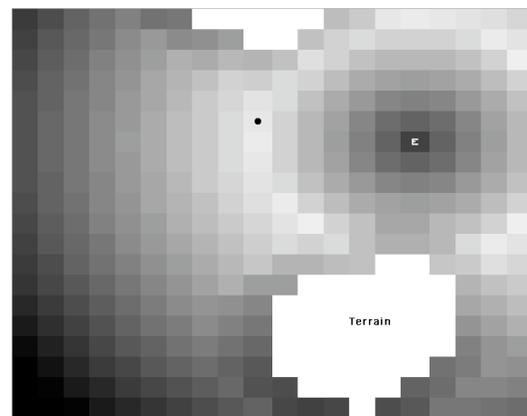

Fig. 2. An own tank (black circle) engaging an enemy unit E. The most attractive potentials are in a circle surrounding the enemy at maximum shooting distance of the tank.

The tactical field is generated by own units, own bases and sheep. These objects generate small repelling fields to avoid our agents from colliding with each other or bases as well as avoiding the sheep. There is also an interesting behaviour emerging from the combination of the strategic field and the tactical field. While the former makes the units keep a certain distance to the opponent units, the latter make them keep a short distance to their own units. Combined, it makes the units able to surround an opponent (as seen in Figures 4–5) without any need of an explicit control to do so.

Each subfield is weighted and summed to a major field, and the major fields are in turn weighted and summed to form a total potential field which is used by our agents for navigation. We will illustrate how the total potential field can look like with an example. Figure 3 shows a potential field view of a worker unit moving from a base to a mine to gather resources. The mine is the goal of the unit and therefore generates an attracting field (lighter grey areas have higher potentials than darker grey areas). The unit must also avoid colliding with obstacles, and the base and terrain therefore generate small repelling fields.
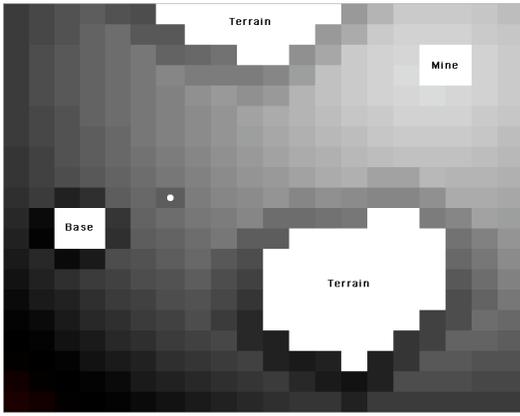
Fig. 3. A worker unit (white circle) moving towards a mine to gather resources. The mine generates an attractive field and the base and terrain generate small repelling fields for obstacle avoidance. Light grey areas are more attracting than darker grey areas. White areas are impassable tiles.

Figures 4 and 5 show a 2D debug view from the game server and the corresponding potential field view during an ORTS tankbattle game. It illustrates how our own units cooperate to engage and surround the enemy at maximum shooting distance. For the interested reader we refer to the original description for more details of the MAPF technology [9], [10].
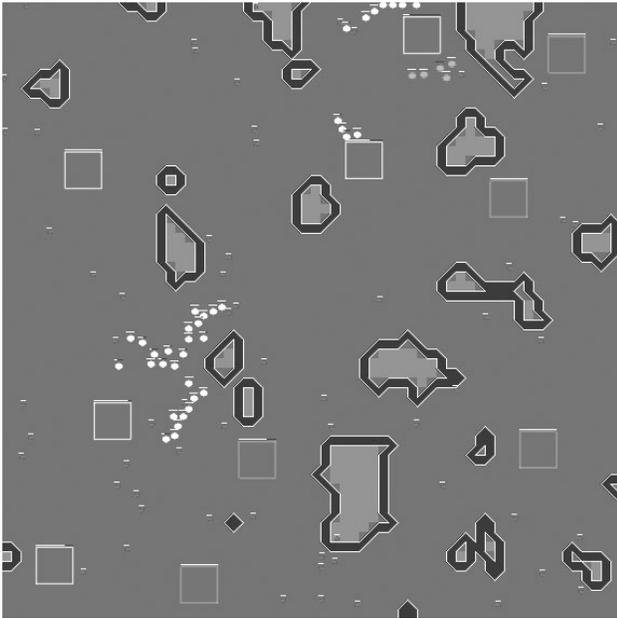

Fig. 4. The 2D view of an ORTS Tankbattle game. Our own tanks (white circles) are engaging the enemy units (grey circles) and bases (grey rectangles).

III. Experimental Setup

The experiments were carried out during DreamHack Winter 2008, the largest LAN party in the world. We were positioned in the boot of our University, where players who stopped at our boot were asked if they would like to
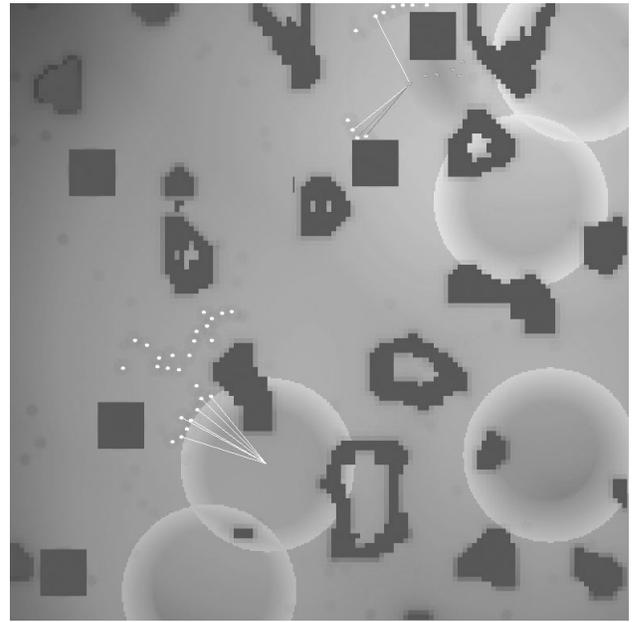

Fig. 5. The potential field view of the same ORTS Tankbattle game. Lighter grey areas have more attracting potentials than darker areas. The white lines illustrate the coordinated attacks on a base (lower left) and a unit (upper right).

participate in a scientific experiment. Those who agreed were given instructions on how to play the game and then played a short session of *ORTS Tank Battle* against one of the bots. After the match, the participants filled in a questionnaire.

A. The Different Bots

We used five different bots, each one based on the MAPF-bot described in previous work [10]. The only changes made were that each unit's ability to act in each time frame was restricted through a probabilistic filter in the following way:

- A bot strength, $s \in [0, 1]$, is initialised in the beginning of the game.
- In each time frame and for each unit, generate a random number $r \in [0, 1]$.
- If $r > s$, let the unit be idle for this time frame.
- If $r \leq s$, let the unit act as usual.

The only thing that we varied in our experiments was the value of s. Two of the bots used constant values of s, while three versions used dynamically adjusted values. The five bot versions are:

Bot A: Static bot with medium difficulty. s is set to 0.3 and does not change during the game.

Bot B: Static bot with low difficulty. s is set to 0.1 and does not change during the game.

Bot C: Adaptive bot. s is initially set to 0.4 and is adjusted during the game using the adaptive difficulty algorithm described below. Learning rate is set to 0.01.

Bot D: Same as Bot C, but when the human player has five or less tanks left s drops to 0.02 to always let the player win.

Bot E: Adaptive bot with high learning rate to quickly adapt to changes in the game. s is initially set to 0.4 and is updated with learning rate 0.2.

B. Adaptive difficulty algorithm

In the adaptive bot versions s is adjusted throughout the course of a game using an adaptive difficulty scaling algorithm. An evaluation function is first used to calculate the relative strength, $isScore$, between the computer and human player at a given time frame. A score above 0 means that the human player is in the lead, and below 0 that the computer player is in the lead. The $aimScore$ is the relative strength the bot aims for, in our experiments 0. By changing the $aimScore$ the difficulty of the adaptive AI is adjusted. By, for example, setting the value to slightly above 0 the goal of the difficulty scaling algorithm is to let the human player be in a small lead. A high positive value will in turn create a computer opponent that is easily beaten. The approach of observing the relative strength between the players were also used by Bakkes et al. in [6]. The difference compared to our approach is in the features involved in the calculation of the relative strength.

The pseudocode below describes how the evaluation score is used to update s. Note that the implementation makes sure s always is between 0.05 and 1.0, except for late in a game for Bot D where s is set to 0.02 bypassing the algorithm. The

---

Algorithm 1 The algorithm that updates the s parameter.

$isScore = 0$
$aimScore = 0$
for all EnemyUnit eu **do**
　$isScore = isScore + 1 + eu.getHealthPercent() * 0.5$
end for
for all OwnUnit ou **do**
　$isScore = isScore - 1 - ou.getHealthPercent() * 0.5$
end for
$diffScore = isScore - aimScore$
$s = s + learningRate * diffScore$
if $s < 0.05$ **then**
　$s = 0.05$
end if
if $s > 1.0$ **then**
　$s = 1.0$
end if

---

adaptive difficulty algorithm was first evaluated against two static bots from the 2007 years' annual ORTS tournament. The purpose was to make sure the difficulty scaling worked correctly before performing experiments with human players. The results from the evaluation is presented in Table I. They show that the scaling, although not entirely perfect, works well enough. The adaptive difficulty bot does not have to win exactly 50% of the games. The goal is rather that it reacts to the opponent and tries to play an even game, which is confirmed by the experiments.

TABLE I
WIN PERCENTAGE OF OUR BOT, BOTH WITH AND WITHOUT ADAPTIVE DIFFICULTY SCALING, AGAINST TWO BOTS FROM 2007 YEARS' ORTS TOURNAMENT.

	Wins (over 100 games)	
Opponent	No Scaling	Adaptive Difficulty
NUS	100%	60%
Uofa06	100%	52%

C. The Questionnaire

The Questionnaire consisted of a brief instruction on how to fill in the answers, a number of general questions (concerning age, sex, and the results of the match, which were filled in by the test leader), and a word pair section.

Before the creation of the questionnaire, we identified three aspects of the opponent bot that we would like to test:

- The experienced *enjoyment* of facing this opponent bot,
- the experienced *strength* of the opponent bot, and
- the experienced *variation* of the opponent bot.

We used six clusters of words, with four words in each cluster (in total 24 words) to describe the bot along three different dimensions. In the questionnaire, the participants had to compare twelve pairs of words, where words from each cluster was paired with words from each other cluster except the opposite one. *Hard, Demanding, Challenging,* and *Impossible* was for instance compared to *Fun, Painful, Monotonous* and *Alternating*, rather than the words in the cluster of *Uncomplicated, Easy, Simple* and *Problem free.* One can think of it as the six clusters being the six sides of a cube. Each side shares four edges, on with every other side, except the opposite side. Each edge at that cube then correspond to a pair of words in our experiment.

Each comparison as such was carried out by the user through marking one of seven options along an ordinal scale, where the center option indicated that both words described the bot equally good (or bad), and the others ranged from *somewhat better* to *much better* in favour of the closest word.

The words used for the different axis are presented in Table II. Since the experiments were carried out in Sweden, we used a Swedish questionnaire (therefore the original words of the questionnaire are provided as well).

The participants did of course not know what version of the bot they faced in their experiment, but we did inform them that it was an experiment aimed at trying to measure different experienced properties of the computer opponent. In total 60 persons participated, evenly distributed among the five different versions of the bot.

IV. THE RESULTS

Since the options in each decision in the questionnaire were formulated in terms of which word that best describes the bot, we do not take into account the possibility that a participant use the scale to negate a property by giving high marks to the other word. Only the cases where the participant decide in favour of a word will count for that cluster, as illustrated in Table IV.

TABLE II
A LIST OF THE WORDS USED IN THE QUESTIONNAIRE. WORD 1 AND
WORD 2 OF THE SAME ROW IN THE TABLE WERE ALSO COMPARED IN
THE QUESTIONNAIRE.

Word 1	in Swedish	Word 2	in Swedish
Hard	Svår	Fun	Kul
Unexpected	Oväntad	Frustrating	Frustrerande
Uncomplicated	Okomplicerad	Variated	Varierad
Painfull	Plågsam	Demanding	Krävande
Predictable	Förutsägbar	Tiresome	Tröttsam
Impossible	Omöjlig	Alternating	Omväxlande
Tedious	Enformig	Easy	Lätt
Surprising	Överraskande	Entertaining	Underhållande
Monotonous	Monoton	Challenging	Utmanande
Enjoying	Roande	Problem free	Problemfri
Captivating	Fängslande	Routine	Rutinmässig
Simple	Enkel	Irritating	Irriterande

TABLE III
THE CLUSTERING OF WORDS IN THE SIX CLUSTERS.

Enjoyment+			
Entertaining	Fun	Enjoying	Attractive
Enjoyment-			
Painfull	Tiresome	Tedious	Irritating
Strength+			
Hard	Demanding	Impossible	Challenging
Strength-			
Uncomplicated	Easy	Problem free	Simple
Variation+			
Unexpected	Variated	Alternating	Surprising
Variation-			
Frustrating	Predictable	Monotonous	Routine

Utvärdering av Spel-AI

Instruktioner

Blanketten innehåller en rad ord som kan användas för att beskriva motståndaren i spelet. Kryssa för en av cirklarna mellan varje ordpar för att markera vilket ord som du tycker bäst beskriver motståndaren i den match du nyss spelat.

Exempel

Om du tycker att ordet FANTASIFULL passar *något* bättre än ordet TRÅKIG, så ska ditt svar se ut så här:

FANTASIFULL ○ ○ ⊗ ○ ○ ○ ○ TRÅKIG

Anser du istället att ordet FANTASIFULL passar *mycket* bättre än ordet TRÅKIG, så ska ditt svar se ut så här:

FANTASIFULL ⊗ ○ ○ ○ ○ ○ ○ TRÅKIG

Markera cirkeln i mitten om du anser att båda orden beskriver motståndaren lika mycket (eller lika lite):

FANTASIFULL ○ ○ ○ ⊗ ○ ○ ○ TRÅKIG

Observera att det finns inga markeringar som är mer rätt"eller fel"än de andra. Det är viktigt att du svarar så ärligt som möjligt.

Enkäten

Kryssa för en av cirklarna mellan varje ordpar för att markera vilket ord som du tycker bäst beskriver motståndaren i den match du nyss spelat.

SVÅR	○ ○ ○ ○ ○ ○ ○	KUL
OVÄNTAD	○ ○ ○ ○ ○ ○ ○	FRUSTRERANDE
OKOMPLICERAD	○ ○ ○ ○ ○ ○ ○	VARIERAD
PLÅGSAM	○ ○ ○ ○ ○ ○ ○	KRÄVANDE
FÖRUTSÄGBAR	○ ○ ○ ○ ○ ○ ○	TRÖTTSAM
OMÖJLIG	○ ○ ○ ○ ○ ○ ○	OMVÄXLANDE
ENFORMIG	○ ○ ○ ○ ○ ○ ○	LÄTT
ÖVERRASKANDE	○ ○ ○ ○ ○ ○ ○	UNDERHÅLLANDE
MONOTON	○ ○ ○ ○ ○ ○ ○	UTMANANDE
ROANDE	○ ○ ○ ○ ○ ○ ○	PROBLEMFRI
FÄNGSLANDE	○ ○ ○ ○ ○ ○ ○	RUTINMÄSSIG
ENKEL	○ ○ ○ ○ ○ ○ ○	IRRITERANDE

Svara också på följande frågor:
Min erfarenhet av att spela realtidsstrategispel sedan tidigare är:

OBEFINTLIG ○ ○ ○ ○ ○ ○ ○ MYCKET LÅNG

Jag är:

MAN ○ KVINNA

Min ålder är:

≤ 10 år ○   11 − 15 år ○   16 − 20 år ○   21 − 25 år ○   26 − 30 år ○   ≥ 31 år ○

Löpnummer: 1   Resultat: Spelaren ☐☐   Boten ☐☐   Botversion A

Fig. 6. The questionnaire used in the experiments. In addition to the instructions, an example of how to fill it in, and the word pair questions, we also asked about the player's age and sex, and asked them to grade their previous experience from RTS games. We also registered the results (in terms of the scores of the players) of the game.

TABLE IV
THE DISTRIBUTION OF SCORES GIVEN A PARTICIPANT MARK
COMPARING *Hard* AND *Fun*. THE SCORES AS SHOWN BELOW IN THE
LOWER TWO ROWS WERE NOT REVEALED TO THE PLAYERS IN THE
QUESTIONNAIRE.

HARD	○	○	○	○	○	○	○	FUN
Strength+	3	2	1	0	0	0	0	
Enjoyment+	0	0	0	0	1	2	3	

We now compare the different aspects A of each bot B in two ways:

1) First, we compare the average scores of each aspect by looking at the clusters separately (as presented in Table VI), and compiled per aspect (strength, variation and enjoyment) as can be seen in Table V.

2) Second, we remove the values given by the participants (0, 1, 2, or 3) in each question and only count the presence of positive judgements (0 or 1). We then make a comparison between the positive and negative aspects, and make a t-test on the results to test the statistical significance of the differences.

V. DISCUSSION

A. The Methodology

We chose to let the participants only try *one* of the five versions. This choice can of course be criticised, since it has been argued that experiments where you let the participants compare a number of different options give more reliable results [11]. Even though they indeed have a point, we were not at all convinced that we would be able to catch their interest so that they would play two or more games in this noisy, distractive environment. We are still convinced that this was a good decision since the impression that we got at the time of the experiments was that the participants were often visiting the boot in groups, and that many of them were

asking about how long it would take (i.e. *how long will my mates have to wait*). Being able to keep down the time it would take for each participant to finish was therefore an important aspect of the experiment design (unfortunately at the cost of getting direct pair wise comparisons).

B. The Results

Table V and VI show that the adaptive Bots C and E were perceived as most varied (although the difference is not statistically significant). The third adaptive bot, D, had the lowest variation score of all. This bot aims for a balanced game until the end where the performance drops so the own tanks in practice are idle. The low variation score is probably

TABLE V
THE AVERAGE RESULTS OF THE THREE ASPECTS.

Bot	Enjoyment	Strength	Variation
A	-0.03	0.46	-0.69
B	-0.04	-1.6	-0.64
C	0.66	-0.38	-0.03
D	-0.94	-1.21	-1.17
E	0.91	-0.62	-0.17

TABLE VI
THE AVERAGE RESULTS OF THE SIX CLUSTERS.

Bot		E+	E-	S+	S-	V+	V-
A	Avg.	1.33	-1.24	1.65	-1.36	1.42	-1.75
	Stdd.	0.62	0.56	0.61	0.67	0.67	0.85
	n	15	17	17	11	12	24
B	Avg.	1.33	-1.55	1	-2	1.44	-1.79
	Stdd.	0.65	0.82	0	0.94	0.73	0.86
	n	12	11	4	26	16	29
C	Avg.	1.71	-1.64	1.38	-1.81	1.88	-1.57
	Stdd.	0.81	0.67	0.51	0.98	0.86	0.68
	n	24	11	13	16	17	21
D	Avg.	1.5	-2.27	1.33	-2.35	1.22	-2
	Stdd.	0.84	0.65	0.5	0.75	0.44	0.75
	n	6	11	20	9	9	26
E	Avg.	1.71	-1.5	1.42	-1.60	1.75	-1.53
	Stdd.	0.86	0.76	0.67	0.65	0.75	0.72
	n	24	8	12	25	12	17

TABLE VII
PAIRED T-TEST FOR ENTERTAINMENT, STRENGTH, AND VARIATION
(E,S,V) FOR EACH PAIR OF BOTS WHERE THE NUMBERS INDICATE THE
PROBABILITY (IN PER CENT) OF THE SAMPLES COMING FROM THE SAME
DISTRIBUTION WITH THE SAME MEAN VALUES. RESULTS THAT ARE
STATISTICALLY SIGNIFICANT AT THE 5% LEVEL ARE IN BOLD.

Bot	B	C	D	E
A	(67, **0.17**, 93)	(7.6, 31, 27)	(69, **1.6**, 65)	(9.7, 6.8, 46)
B	—	(8.4, **3.8**, 35)	(44, 28, 62)	(9.1, 28, 43)
C	—	—	(5.4, 19, 8.7)	(92, 15, 74)
D	—	—	—	(**2.1**, 100, 19)

due to that players have a tendency to remember the last parts of the game when filling in the forms. Since the bot is very dumbed down during this phase of the game it is perceived as static and boring. Mat Buckland's saying that "it only takes a small dodgy-looking event to damage a player's confidence in the AI" is very applicable to this bot version [12].

The static Bot A was the most difficult to beat (the result is significant at the 5% level compared to Bot B and Bot D). The low performance static Bot B and the adaptive Bot D that drops performance were very easy to beat. The adaptive Bots C and E were somewhere in the middle with some participants perceiving them as relatively easy to beat while others felt they were quite hard to beat, which of course was the intention of using runtime difficulty scaling algorithm.

The adaptive Bots C and E were by far considered most enjoying to play against. The static Bots A and B were scored as enjoying by some participants and not enjoying by others, ending up around a zero score. A majority of the participants that played against Bot D, felt that it was not fun at all to meet.

VI. CONCLUSIONS AND FUTURE WORK

The results show that the adaptive Bots C and E were perceived as comparingly varied, neither being too hard nor too easy to beat, and being the most fun to play against of the versions used in the experiment, although more experiments are needed to support these results at a statistically significant level. It supports our beliefs though that players prefer opponents that models and adapts to the player and that can be beaten if the player plays the game well. It would however be interesting to see how adaptive bots with higher and lower difficulty levels would be perceived.

The static bots were, compared to the adaptive ones, perceived as boring to play against and their difficulty level was not balanced to the player (Bot A was too hard and Bot B too easy to beat). This result is statistically significant at the 5% level when we compare the enjoyment of A+B to the one of C+E. It is often difficult to find a static difficulty level that suit the preferences of many players. The risk with static levels is that one difficulty setting is way too easy for a player while the next level is too hard. For example Warcraft III has been criticised in forums and reviews for being too difficult in skirmish games against computer players [13], [14]. The adaptive Bot D that drops the performance in the end of a game was perceived as boring, easy to win against and with low variation. It is interesting to see that this bot was considered least varied even if it uses the same adaptive technique as Bot C until the performance drops and tanks in practice are almost idle. This might be because participants have a tendency to mostly remember the last part of the game where the AI is really dumb, and therefore consider the bot as too easy and boring.

Future work would include experiments in a more calm and controlled environment. The participants would get some time to play against a default opponent to learn a bit about the game mechanics, and then get to play against one, or a few of the bot versions. The different versions could also be slightly revised. Bot D, the one that drops performance considerably in the end game, is by the participants regarded as not very enjoyable, easy to beat and with least varation.

Another direction could be to use other techniques for compensating the difference in strength between the players. The approach we used was to let units be idle with some probability. Instead the difference can be compensated with changes in higher level tactics, for example switch to a more non-aggresive playing style if the computer player is in lead [6]. Still it is interesting to see that such a simple way of runtime difficulty scaling was clearly perceived as more varied and entertaining than the static bots in the experiments.

VII. ACKNOWLEDGEMENTS

REFERENCES

[1] J. J. Macaloon, *This Great Symbol: Pierre de Coubertin and the Origins of the Modern Olympic Games*, 1st ed. Routledge, November 2007. [Online]. Available: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/041549494X

[2] "DreamHack," http://www.dreamhack.se URL last visited on 2009-08-05, 2009.

[3] T. W. Malone, "Toward a theory of intrinsically motivating instruction," *Cognitive Science*, no. 4, pp. 333–370, 1981.

[4] ——, "What makes computer games fun?" *BYTE*, no. 6, pp. 258–277, 1981.

[5] P. Sweetster and P. Wyeth, "Gameflow: A model for evaluating player enjoyment in games," *ACM Computers in Entertainment*, vol. 3, no. 3, 2005.

[6] S. Bakkes, P. Spronck, and J. van den Herik, "Rapid Adaptation of Video Game AI," in *Proceedings of 2008 IEEE Symposium on Computational Intelligence and Games (CIG)*, 2008.

[7] J. K. Olesen, G. N. Yannakakis, and J. Hallam, "Real-time challenge balance in an RTS game using rtNEAT," in *Proceedings of 2008 IEEE Symposium on Computational Intelligence and Games (CIG)*, 2008.

[8] M. Buro, "ORTS — A Free Software RTS Game Engine," 2007, http://www.cs.ualberta.ca/∼mburo/orts/ URL last visited on 2008-08-26.

[9] J. Hagelbäck and S. J. Johansson, "Using multi-agent potential fields in real-time strategy games," in *Proceedings of the Seventh International Conference on Autonomous Agents and Multi-agent Systems (AAMAS)*, L. Padgham and D. Parkes, Eds., 2008.

[10] ——, "The rise of potential fields in real time strategy bots," in *Proceedings of Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 2008.

[11] G. N. Yannakakis and J. Hallam, "Towards Optimizing Entertainment in Computer Games," *Applied Artificial Intelligence*, pp. 21:933–971, 2007.

[12] M. Buckland, *Programming Game AI by Example*. Wordware Publishing, Inc., 2005.

[13] "Warcraft III Guide," http://guides.ign.com/guides/12906/page_6.html URL last visited on 2009-06-09, 2008.

[14] "Warcraft III Game reviews," http://reviews.cnet.com/pc-games/warcraft
-iii-the-frozen/4505-9696_7-30602384.html URL last visited on 2009-06-09, 2008.