Desirable Behaviors for Companion Bots in First-Person Shooters

Adina Friedman[1] and Jacob Schrum[2]

***Abstract*—First-Person Shooter games are a popular genre that often includes a team deathmatch mode of play: teams of agents score by killing members of the other team. When played without other humans, this mode features both opponent bots and companion bots. This paper uses a human subject study with 30 participants to analyze player preference for cooperative teammates vs. skilled, but less cooperative, teammates in the game Unreal Tournament 2004. Specifically, participants play with both a skilled bot based on neuroevolution and a less skilled bot hand-coded to be more cooperative. Survey results indicate that users perceived significant differences between the bots in several categories, e.g. following behavior and skill at scoring, but did not have a significant preference for one bot over another. However, participants did significantly prefer whichever bot they personally felt was more "helpful" and also preferred whichever bot they happened to see more of. These data indicate how user perception can strongly depend on coincidental interactions, such as being seen more. They also identify some qualities that humans desire in teammates, and indicate that simply scoring more will not necessarily result in a higher user preference.**

I. INTRODUCTION

First-Person Shooter (FPS) games are a popular genre often played online by humans in a multiplayer setting, but also played against and even alongside bots that fill in roles otherwise taken by human players. Much effort has been dedicated to creating bots that can convincingly replace humans in many games [1], particularly in the FPS Unreal Tournament 2004 (UT2004; Figure 1; Section III) [2], [3], [4], [5], [6], [7]. This paper focuses on the problem of assessing what humans want out of a teammate bot (in contrast to an opponent), in hopes of working toward better designs for such cooperative agents.

UT2004 is used to assess human preference for two different companion bots in team deathmatch, where the team with the most kills at the end of the game wins. One might assume that humans prefer whichever teammate scores better, but that assumption is challenged in this paper. One bot was hand coded to directly follow and generally focus on the human teammate when possible, while the other bot's behavior was evolved using a variant of NeuroEvolution of Augmenting Topologies (NEAT [8]) called Modular Multiobjective NEAT (MM-NEAT [9]), and was more proficient at scoring points, but had no obligation to follow or interact with the player.

[1]A. Friedman is an undergraduate attending Southwestern University, Georgetown, TX 78626, USA friedmaa@southwestern.edu

[2]J. Schrum is an Assistant Professor of Computer Science at Southwestern University, Georgetown, TX 78626, USA schrum2@southwestern.edu

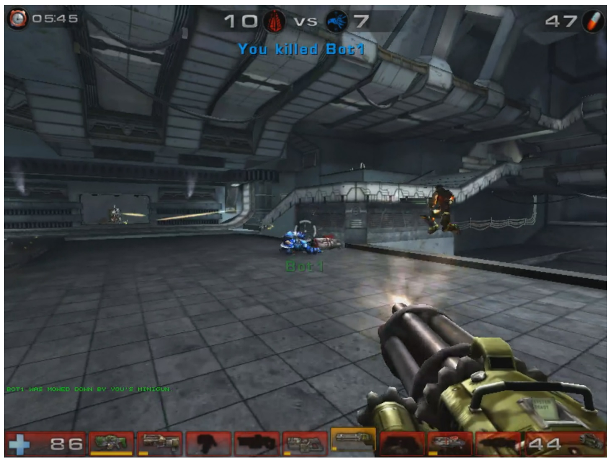


Fig. 1. **Human Subject Playing Unreal Tournament 2004.** User killing native bot with the help of bot Ethan (Section III-B).

Thirty participants played rounds of team deathmatch with each bot, chose which bot they preferred, and provided additional feedback via surveys. Bots were rated in terms of how well they followed, how helpful they were, how often they were seen, how much they scored, and how well they avoided death. The hand-coded bot was better at following which may have led to it being seen more often. The evolved bot scored better and avoided death better. However, the evolved bot was also rated as more helpful, which begs the question of how users interpreted this label (Section V).

In general participants preferred the bot they believed they saw more often, and the bot they perceived as being more helpful, regardless of which bot this was. Additionally, players' open ended feedback showed a clear trend in the reasons behind their preferences. Players who preferred the hand-coded bot tended to prioritize the feeling of teamwork and the game experience more than points. In contrast, players who preferred the evolved bot more often cited its ability to score points and avoid death as the reasons for their choice.

These findings show that player preference cannot be predicted based only on the outcome of a game. Players favored bots based on what they expected to get out of the video game experience more so than simply on the outcome. Players did not universally favor the bot they scored higher with, and they did not significantly favor either bot over the other. However, they did significantly favor interaction (seeing the bot more) and helpfulness, which gives insight into how companion bots should be designed in the future.

The paper proceeds by covering related work in Section II.

UT2004 and the bots made for team deathmatch are described in Section III. The human subject study and its results are described in Section IV, followed by discussion and future work (Section V), before concluding (Section VI).

II. Related Work

Companion bots are an important feature in many commercial games, with some prominent examples being the Mass Effect and Fallout series, Bioshock Infinite, and Metal Gear Solid V. Therefore, researchers have also put effort into the development of companion bots with desirable features.

Motivated Reinforcement Learning has been used imbue bots with a sense of curiosity in open-ended worlds [10]. The concept of Coupled Empowerment Maximisation has been applied to a minimal 2D dungeon crawler scenario to create companion bots [11]. Work in a simple 2D shooting game with some predator/prey elements has focused on online adaptation of teammate behavior in the face of adaptive opponents [12]. Several other examples of teammate AI are discussed in a survey by McGee and Abraham [13].

A related area of research is work focused on developing human-like behavior for bots [1], [14]. Many humans prefer human teammates because of limitations of bots, so research focused on overcoming these limitations and making bots appear more human could also lead to better companion bots.

The game Unreal Tournament 2004 (Section III) is significant for the study of human-like bot behavior because it was used from 2008 to 2012 in the BotPrize competition [2], [3]. BotPrize was a *Turing Test* [15] for bots, requiring victors to be rated as human over 50% of the time. In 2008 and 2009, bots played several three-way deathmatches against different human judges and confederates [2], and judges classified each opponent as human or bot after each match. From 2010 to 2012, bots were instead tested in matches against all other bot entrants and an equal number of human judges [3]. In this variant, human judges tagged combatants as *Human* or *Bot* using a special judging gun. At the end of the match, bots received humanness ratings equal to the percentage of times they were judged as human out of all their judgments.

Entrants made gradual progress toward breaking the 50% humanness barrier from year to year until two bots were victorious in 2012: MirrorBot [4] and UT^2 [7]. MirrorBot mimics the behavior of individuals it interacts with. UT^2 made use of both human trace data [6] and an evolved neural network combat controller [5]. The evolution approach used by UT^2 is similar to the approach applied by one of the bots in this paper (Section III-B). Both bots used in this paper will be discussed after some additional background on UT2004.

III. Companion Bots in UT2004

Unreal Tournament 2004 (UT2004) is an FPS released by Epic Games. One game type it supports is deathmatch: a free-for-all in which players respawn after dying, and points are earned by killing other players. Team deathmatch is an extension of regular deathmatch, where players kill members of the opposing team with the help of teammates, and victory is determined by the collective team score.

Bots can be designed for UT2004 using the Java middleware Pogamut [16], which enables control of virtual agents by allowing them to execute complicated tasks such as pathfinding, information gathering, and memory access using simple commands. Though native bots in many games are able to see the entire map and player positions even when they are not visible, Pogamut only allows bots to to collect and store information within their field of vision, replicating human awareness. Pogamut communicates with UT2004 using GameBots, a mod that allows for control of agents in UT2004. Pogamut made the aforementioned BotPrize competition possible, and is also used to design the bots studied in this paper.

Bots named Jude and Ethan were developed for this study. Jude is a hand-coded bot of modest skill designed to follow members of its team. Ethan is controlled by an evolved neural network, and is better at killing enemies, but is less cooperative. An ideal companion bot would be both cooperative and highly skilled, but to find which focus players preferred, we created two extremes. Jude emphasizes cooperative behavior, and Ethan emphasizes skill. All source code for these bots and the subsequent human subject study (Section IV) is available as part of the MM-NEAT code repository[1].

A. Designing Jude

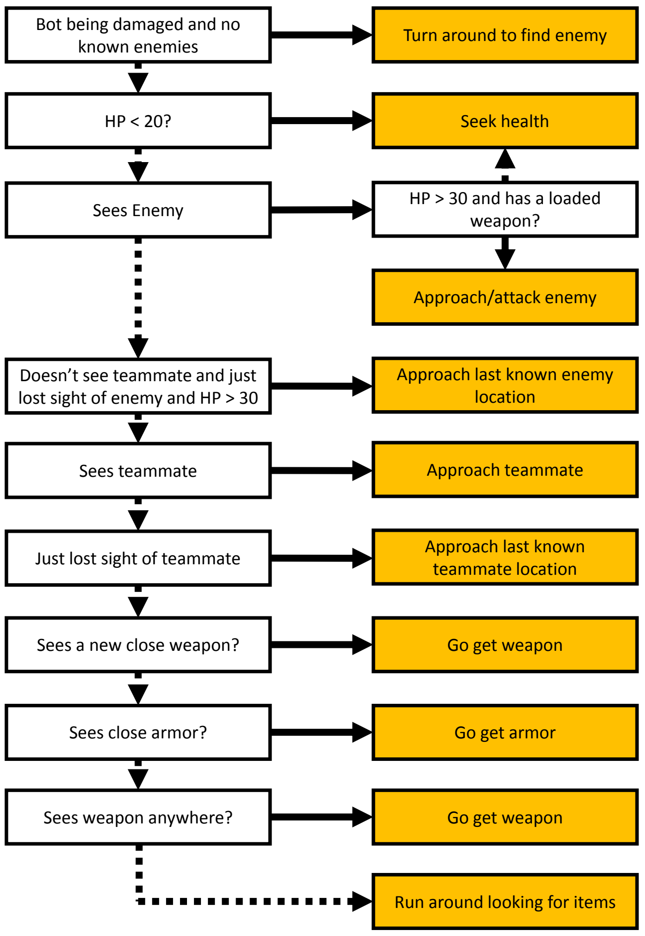
Jude's architecture is in Fig. 2. Jude's behavior mimics human players' tendency to stay close together to more easily defeat enemies. However, Jude also prioritizes its own health, both for the sake of survival, and to assure that human teammates are not overburdened with protecting Jude.

Jude uses a priority list of behaviors. If Jude is being attacked and cannot identify the source, Jude turns to find the enemy. Otherwise, if it has under 20 hit points (HP) out of a possible 100, it heads to the nearest health pickup. If Jude sees an enemy, it only engages if it is sufficiently healthy (over 30 HP) and possesses a loaded weapon, otherwise it seeks a health item. If Jude does not see any agents (friend or foe), but recently saw an enemy (within 1000 game ticks), it will pursue the foe by moving to its last known location.

If there are no visible enemies, Jude is not being damaged, and it sees a teammate, Jude follows the teammate. If Jude loses sight of the teammate, it will go to the teammate's last known location to try to see them again. This simple collaborative behavior causes Jude to often fight alongside human teammates against enemies. When Jude is alone in an area of the map with no visible enemies or teammates, it will run around picking up useful items, prioritizing nearby weapons and armor, which allows it to explore the map and often leads to it encountering another agent.

Thresholds for health checks were based on experience playing UT2004. Players are close to dying if their health is under 20, and they can easily die from a single hit if they enter combat with only 30 HP.

[1]https://github.com/schrum2/MM-NEAT



Bot being damaged and no known enemies
Turn around to find enemy
HP < 20?
Seek health
Sees Enemy
HP > 30 and has a loaded weapon?
Approach/attack enemy
Doesn't see teammate and just lost sight of enemy and HP > 30
Approach last known enemy location
Sees teammate
Approach teammate
Just lost sight of teammate
Approach last known teammate location
Sees a new close weapon?
Go get weapon
Sees close armor?
Go get armor
Sees weapon anywhere?
Go get weapon
Run around looking for items

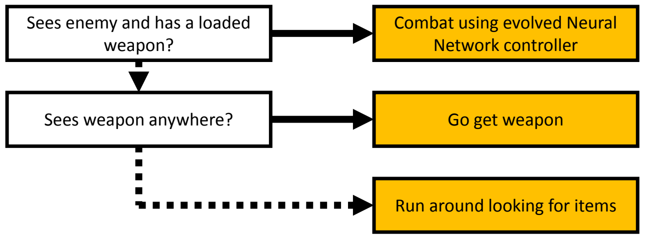
Fig. 2. **Agent Architecture for Jude.** White conditional checks (mostly on the left) are evaluated from top to bottom. When a test results in "Yes" the bold arrow is followed. Otherwise the dotted arrow is followed. Yellow boxes contain actions that are executed. Items are "close" if within 300 UT units.

When both an enemy and a teammate are visible, the bot prioritizes enemy interaction over team unity, since the bot can be easily killed if it prioritizes the player's safety over its own. Therefore Jude will keep itself alive in a fight rather than sacrificing itself for a teammate because this damages the team score. However, Jude's actual combat behavior is purposefully simplistic: Jude heads directly at the enemy shooting its weapon, and does not dodge or jump. To some extent, this means Jude serves as a decoy for the human teammate.

This architecture creates a competent companion that prioritizes teamwork. The strategy of attacking opponents head-on can work with the right weapon, if the opponent is sufficiently weak, but also makes Jude vulnerable to counter-attack. A higher score can be achieved by optimizing the combat behavior, as is demonstrated by Ethan.

B. Evolving Ethan

Ethan's agent architecture is simpler (Figure 3), but has more sophisticated combat behavior because these actions are dictated by an evolved neural network that always engages enemies. Ethan is a streamlined version of UT^2 [5] with tweaks



Sees enemy and has a loaded weapon?
Combat using evolved Neural Network controller
Sees weapon anywhere?
Go get weapon
Run around looking for items

Fig. 3. **Agent Architecture for Ethan.** Same format as Figure 2.

for human-like play (e.g. human trace data [6]) removed. Ethan was evolved from scratch for this paper using a variant of NEAT [8] called MM-NEAT (Modular Multiobjective Neuroevolution of Augmenting Topologies [9]).

NEAT is a neuroevolution algorithm that generates neural networks with arbitrary topologies, and has been applied to many game domains [17], [5], [9], [18]. NEAT networks start simple (no hidden neurons) and complexify over generations via structural mutations that splice in extra neurons along existing links and add new links. Link weights are also perturbed via mutation, and historical markers for each gene make aligning different topologies for crossover efficient and meaningful. MM-NEAT adds support for multiple objectives and modular networks, though modular networks are not used in this paper. Support for multiple objectives is accomplished by using Non-dominated Sorting Algorithm II (NSGA-II [19]), a popular multiobjective evolutionary algorithm.

Ethan was evolved to maximize team score and damage dealt by team members, while minimizing damage received by team members. Although score is all that matters, the other objectives provide smoother gradients than team score. Multiple objectives are therefore useful, even though these objectives are not necessarily contradictory. Objectives focus on the whole team because each evaluation consists of two copies of the same neural network on one team competing against two native bots on the opposing team.

Evaluations in UT2004 are costly. Two minute matches were used to provide sufficient time for interaction, and networks were evaluated three times each to account for different outcomes on each evaluation (noise). Actual scores in each objective were averages across the three trials. The level `DM-Flux2` was used because it is relatively small, so agents frequently interact, but has enough partitioned areas that teammates must search to find each other. A small population size of 10 (NSGA-II generates 10 children from 10 parents on every generation) was evolved for 100 generations. Additional parameters include: crossover rate of 50%, new link rate of 40%, node splice rate of 20%, and a per-link perturbation rate of 5%. Despite the small population size, networks quickly evolved to a point where they could consistently beat the native bots and were playing at a skill level comparable to that of a moderately skilled human. The champion with the best final team score was used in the human subject study.

This success depends on a model of how to derive neural network inputs and process neural network outputs that borrows heavily from UT^2 [5], [7], which was in turn based on earlier work [20]. However, for each sensor that provides information about enemy agents, there is now a corresponding sensor for that same information about teammates. The network processed the following inputs on each game tick:

- 12 ray trace sensors to detect distances to level obstacles, like walls. The ray traces run parallel to the ground and are evenly spaced radially around the bot.
- Ray trace aimed straight ahead that responds to enemies.
- Two sets of 10 pie slice sensors that are used to sense nearby agents, with the value of each slice's input being higher if an agent is closer (Figure 4). One set of 10 sensors is for the teammate, and the other is for enemies. If both enemies are present in one slice, the distance to the closest enemy determines the input value.
- Scaled distance to the nearest enemy and teammate both in 3D and projected onto the 2D plane of the ground. Max distance is 1000 UT units (an arbitrary in-game measurement) and absent agents are interpreted as having the maximal distance. However, additional sensors track this information even for absent teammates, similar to how UT2004 provides players with text descriptions of where teammates are within a map.
- Scaled 3D distances to the nearest enemy and teammate to the left and right of where the bot is looking. These more specific sensors indicate whether the relevant agent is to the left or right of where the bot is looking.
- Is nearest opponent still? Nearest teammate?
- Is nearest opponent shooting? Nearest teammate?
- Is nearest opponent jumping? Nearest teammate?
- Scaled armor points out of 100.
- Scaled health points out of 100.
- Is the bot touching the ground?
- Is the bot being damaged?
- Is the bot bumping into an agent or the environment?
- Is the bot damaging another player?
- Is the bot colliding with the environment?
- Did the bot just fall off a cliff?
- The maximum/minimum/average teammate health (all the same with only one teammate).

This collection of 60 inputs is fed into the neural network and processed to generate 4 output values that define the action of the agent. Since the neural network controller is only used when an enemy agent is visible, all actions are defined with respect to the nearest enemy the bot can see. Specifically, one output is a forward/backward impulse, where positive/negative outputs correspond to moving toward/away from the nearest opponent. The next output helps with dodging, since it is a left/right impulse that interprets positive/negative outputs as the degree to which the bot should strafe sideways in the corresponding direction with respect to the opponent. Finally, there is an output that makes the bot shoot on positive outputs, and an output that makes the bot jump on positive outputs.

The evolved bot was a skilled killing machine in the human subject study described next.

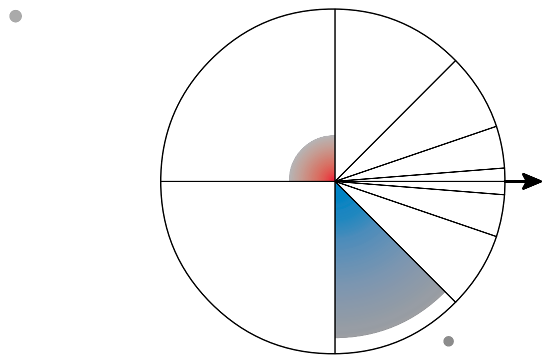


Fig. 4. **Pie Slice Sensors.** There are more slices of smaller size near the front (right) so the bot can better distinguish locations in front of it. Dots represent agents, and filled portions of pie slices show the relative activations for agents at different distances. Activation increases as agents get nearer, so the agent in the upper left causes the corresponding pie slice to be filled less than the slice for the nearer agent on the lower right. The bot has one set of sensors for enemies and one for teammates. Adapted from [20], [5].

IV. Human Subject Study

This section discusses the procedure for the human subject study, and describes its quantitative and qualitative results.

A. Procedure

The human subject study consisted of 30 participants (students, faculty, and staff at Southwestern University). To ensure that results were not biased by the order in which participants encountered the bots, half played with Jude first, while the other half played with Ethan first. All matches took place in `DM-Flux2`, the same level that Ethan was evolved in.

Participants first played a one-on-one tutorial match against a native bot for five minutes with an investigator present to explain the controls and game mechanics. This tutorial match assured some competency and familiarity with the game.

Afterward, the investigator would leave the room and the participant would play a 10 minute round of team deathmatch against two native bots with either Jude or Ethan as a teammate. Participants would then take a brief survey before playing another 10 minute round with the other teammate. Participants then filled out another survey about their experience and their preferences between the two bots. In addition to the survey data, play sessions were recorded using screen capture software. These recordings are available online at southwestern.edu/~schrum2/SCOPE/ut2004-companions.php.

After each round the players were asked what qualities they liked and disliked about the bot, and what changes they would recommend to improve the bot's behavior. Players rated the bots on a scale from 1 to 5 on five different metrics, with a higher score meaning that they considered the bot to be better in that aspect. The bots were scored on how well they followed the player, how helpful they were, how often the player saw them, their ability to score points, and how well they avoided dying. Participants also indicated which bot they preferred playing with (referred to in the survey as "The First Bot" and "The Second Bot") and the reasons for that preference.

TABLE I
User Ratings for Bots

	Jude						Ethan						
	1	2	3	4	5	Avg.	1	2	3	4	5	Avg.	p
Followed Well	5	8	6	9	2	$\mathbf{2.8\overline{3}}$	10	11	6	2	1	2.1	**0.01834834**
	$16.\overline{6}\%$	$26.\overline{6}\%$	20%	30%	$6.\overline{6}\%$		$33.\overline{3}\%$	$36.\overline{6}\%$	20%	$6.\overline{6}\%$	$3.\overline{3}\%$		
Was Helpful	1	5	9	14	1	3.3	2	0	5	7	15	**4.14**	$\mathbf{4.77 \times 10^{-4}}$
	$3.\overline{3}\%$	$16.\overline{6}\%$	30%	$46.\overline{6}\%$	$3.\overline{3}\%$		6.89%	0%	17.24%	24.13%	51.72%		
Seen Often	0	3	12	7	8	$\mathbf{3.\overline{6}}$	2	7	12	7	2	3	**0.02051698**
	0%	10%	40%	$23.\overline{3}\%$	$26.\overline{6}\%$		$6.\overline{6}\%$	$23.\overline{3}\%$	40%	$23.\overline{3}\%$	$6.\overline{6}\%$		
Scored Well	0	5	10	12	3	$3.4\overline{3}$	1	1	3	6	19	$\mathbf{4.3\overline{6}}$	$\mathbf{6.23 \times 10^{-5}}$
	0%	$16.\overline{6}\%$	$33.\overline{3}\%$	40%	10%		$3.\overline{3}\%$	$3.\overline{3}\%$	10%	20%	$63.\overline{3}\%$		
Avoided Death	4	11	12	2	1	2.5	1	2	7	6	14	**4**	$\mathbf{1.76 \times 10^{-6}}$
	$13.\overline{3}\%$	$36.\overline{6}\%$	40%	$6.\overline{6}\%$	$3.\overline{3}\%$		$3.\overline{3}\%$	$6.\overline{6}\%$	$23.\overline{3}\%$	20%	$46.\overline{6}\%$		

Number of participants who gave Jude and Ethan specific ratings in each category on a scale from 1–5, including associated percentages. Average ratings are also shown, with **bold** scores indicating which bot earned the higher score in each category. The p-values of statistical comparisons using two-tailed Wilcoxon-Mann-Whitney U Tests are also shown, which are significant in each case. To account for the high number of ties in the data, the EDISON-WMW [21] algorithm was used to compute precise p-values.

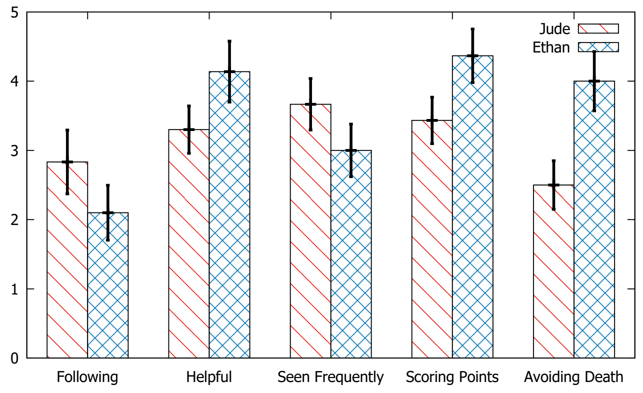


Fig. 5. **Average User Assessments of Both Bots.** Scores on a 1-5 scale are shown side-by-side for Jude and Ethan, along with 95% confidence intervals.

B. Quantitative Results

On average, players contributed 43.09% of the points in matches with Jude with an average team score of 25.71. With Ethan, players contributed 34.53% of the points with an average team score of 33.75. According to two-tailed Wilcoxon-Mann-Whitney U Tests, the team scores with Ethan are significantly higher ($p \approx 1.027 \times 10^{-4}$), but the percentage of points contributed by humans was significantly higher with Jude ($p \approx 0.0115$). In fact, for 18 out of 28 observed players, the percentage of team points from the human was higher with Jude, while the percentage for the remaining 10 was higher with Ethan. These data are only based on 28 participants because the videos for two participants were recorded incorrectly, making the scores in these matches uncertain.

Participant ratings of Jude and Ethan are in Table I. The number of individual ratings for each value from 1 to 5 are shown, along with percentages. Because one user neglected to answer the question of how helpful Ethan was, these percentages are only calculated out of 29 instead of 30. Average ratings are provided in the tables as well, along with the results of two-tailed Wilcoxon-Mann-Whitney U Tests comparing each bot. The differences are statistically significant in each case ($p < 0.05$). These averages are also depicted visually in Figure 5 with 95% confidence intervals. Specifically, users rated Jude as better at following and more frequently seen. Users rated Ethan as being more helpful, and as being better at scoring and avoiding death. Most of Jude's ratings were around 3 and spread out from 1–5, though no one gave Jude a score of 1 for being seen often or for scoring well. Ethan's scores skew more strongly in one direction or the other, with ratings for following on the low end, and ratings for helpfulness, scoring, and avoiding death all skewing toward the high end. Only the score for being seen often is middling, with an average of 3. Although Ethan's helpfulness score skews toward the high end, and 0 respondents gave Ethan a rating of 2 in this category, two respondents did rate Ethan as a 1 for helpfulness.

TABLE II
Relative User Ratings for Bots

Better at ...	Jude	Ethan	Tie	p
Following	$53.\overline{3}\%(16)$	30%(9)	$16.\overline{6}\%(5)$	0.3616
Helping	17.24%(5)	68.97%(20)	13.79%(4)	**0.00813**
Being Seen	$43.\overline{3}\%(13)$	$6.\overline{6}\%(5)$	40%(12)	0.2005
Scoring	$6.\overline{6}\%(2)$	60%(18)	$33.\overline{3}\%(10)$	**0.005223**
Avoiding Death	$6.\overline{6}\%(2)$	$63.\overline{3}\%(19)$	30%(9)	**0.005223**

Percentages and counts for which bot each user rated higher in each category, and the number of tied ratings. The p-values for the results of a Binomial Test are also shown (ties split between both bots), with significant differences in **bold**.

In addition to looking at raw scores, the data is compared in terms of which bot the participant rated higher in each metric. A certain rating from one participant may not be comparable to the same rating from another, but it is meaningful when a user rates each bot differently in the same category. These comparisons are in Table II. This data is analyzed using a Binomial Test which assumes there is an even split between individuals that rate Jude higher vs. Ethan. However, applying this test is complicated by the presence of actual ties in the ratings. Simply ignoring these ties leads to significant differences being found in more categories. Therefore, a more conservative approach is used: tied scores are split up evenly between Jude and Ethan. If there is an odd number of ties, then the extra point is assigned to whichever of the bots causes the null hypothesis to be less likely to be rejected. The p-values for these tests are also shown in Table II.

Wherever the Binomial Test returns a significant difference, participants rated Ethan higher than Jude. Significantly more individuals found Ethan more helpful, and better at scoring and avoiding death. The number of ties in the categories of scoring and avoiding death were high (10 and 9 respectively), but few participants thought that Jude did better in either of these categories (just 2 in each case). Participants felt Jude was better at following and was seen more often, but not enough to constitute a significant difference. In particular, the category of being seen had 12 ties and 13 individuals that favored Jude. These 13 respondents are more than the 5 individuals that saw Ethan more often, but the many ties cause the null hypothesis (Ethan and Jude seen equally often) to not be rejected.

Users were also asked which bot they preferred: 17 players ($56.\overline{6}\%$) preferred Ethan, and 13 players ($43.\overline{3}\%$) preferred Jude. A Binomial Test on these responses reveals no significant preference for one bot over the other ($p \approx 0.5847$). However, analyzing these responses in conjunction with user ratings of the bots identifies which types of behavior players prefer.

Specifically, responses were analyzed with a Binomial Test to see whether players preferred the bot they rated higher or lower in each category. As above, ties are split as evenly as possible between categories. Table III shows how many users favored the bot that they specifically rated as better or worse in each category, with p-values from the Binomial Tests as well. Users only had a significant preference for the bot they saw more of, and the bot they perceived as helping more. Although more players preferred the bot that they perceived to be a better follower, this preference is not significant, with a whole 7 participants preferring the bot that was worse at following. More players also preferred the bot that was better at scoring and avoiding death, but these preferences are not significant either. In these two cases, 6 individuals preferred the agent that they rated worse in these categories, and there are many ties: 10 for scoring and 9 for avoiding death.

Player preferences were also analyzed with respect to player gender and level of experience with games in general and UT2004 in particular, but the preference for Ethan vs. Jude was not significantly different in any of these subgroups. Regardless of gender or experience, users like helpful bots that they see frequently. However, although following leads to a bot being seen more often, it was not significantly favored. The exact meaning of “helpful” is also uncertain. However, open-ended user responses shed light on these points of uncertainty.

C. Qualitative Results

Players who preferred Jude tended to emphasize the feeling of teamwork over the objective results of its abilities. This is epitomized by one subject who said “With [Jude] it felt like we were a team even though we did not score as many points.” Players said they felt like they were able to employ more team based tactics because they knew that Jude would follow them into battle, and felt that the bot would “cover them” in combat. They also said that Jude felt more like it was helping them, rather than just going off on its own. These comments support the idea that humans might prefer a bot who is more team oriented, even at the cost of reduced performance.

TABLE III
Player Behavior Preferences

Category	Prefer Better	Prefer Worse	Tie	p
Follower	60%(18)	$23.\overline{3}\%$(7)	$16.\overline{6}\%$(5)	0.09874
Helper	65.52%(19)	20.69%(6)	13.79%(4)	**0.02412**
Seen More	50%(15)	10%(3)	40%(12)	**0.04277**
Scored	$46.\overline{6}\%$(14)	20%(6)	$33.\overline{3}\%$(10)	0.2005
Avoided Death	50%(15)	20%(6)	30%(9)	0.2005

Percentages and counts for users who preferred the bot that they personally rated better/worse in each category, with ties shown as well. Scores for the two bots in the same category (Table I) were combined with each user's stated bot preference to determine if they prefer high or low scores in the category. The p-values for the results of a Binomial Test are also shown (ties split between both bots), with significant differences in **bold**.

Players who preferred Ethan tended to justify their preference based on Ethan's proficiency at scoring and avoiding death, both of which affected the final outcome of the game. Players remarked that Ethan died less than Jude which made some feel like it was less of a burden. They also commented that Ethan was better at killing enemies and more proficient in combat, hence better at scoring. One player said that they preferred Ethan because it “made the game easy to win” while another remarked that if they wanted a more challenging experience, Jude might be a better choice. Players who preferred Ethan prioritized the final output of the game over the experience itself, or were the sort of player who preferred to act independently of a teammate rather than collaboratively.

However, Ethan's proficiency also had some negative impacts on the player experience. Multiple players commented that they felt ignored by Ethan, with one saying that Ethan “was too good. I did not feel as though we were a team.” In total, five players expressed similar sentiment, and all but one preferred Jude. Players also complained that they felt their participation was not needed, with two commenting that the bot would steal kills from the player which was frustrating.

This feeling that the bot was too powerful led players to feel that Ethan could win the match without them, which they found discouraging. Four players commented that Ethan made the game too easy, and removed the challenge of the game.

One participant summed up the reason behind these sorts of feelings towards Ethan's proficiency perfectly by saying that “As counter-intuitive as it might seem, players often enjoy an ally that is good, but not quite as good as the player themselves. Players enjoy ‘being the hero’ of the game.”

The perception of Ethan as a worse follower is also evident in the open-ended feedback, as 14 participants, including ones who ultimately preferred Ethan over Jude, suggested that Ethan could be improved by following the player more. This recommendation suggests that even though players may like having a bot that is able to act on it's own, they still value close proximity to teammates, because it fosters teamwork.

Multiple players remarked on Ethan's propensity to jump even when it was not strictly necessary, and many suggested this was a bug in its programming, but this sort of movement is not unusual in human players. From a tactical standpoint, jumping makes a player harder to hit, which is a benefit in games like UT2004 where players want to avoid being killed. It is therefore not surprising that this behavior evolved.

Some players complained that Jude was too aggressive and would often die because it would prioritize combat over staying alive. This observation is not surprising given how simplistic Jude's combat behavior is. Jude simply rushes straight at the opponent, which is aggressive and makes Jude easier to kill. Users also complained that the bot would try to fight both enemies alone rather than trying to find its teammate, which resulted in it being killed more often.

Multiple users complained that both bots would get stuck on parts of the environment and could not free themselves either for an extended period of time or until they were killed. This outcome is due to Pogamut, which despite using the same navigation grids as native UT2004 bots, does not follow paths and avoid obstacles as well as native bots. In fact, avoiding getting stuck and getting unstuck were major challenges that UT^2 had to be overcome to succeed in BotPrize [7].

The quantitative results indicate that Jude is the better follower, and the open-ended responses indicate that most players took this as a positive trait. However, some complained that Jude followed them too closely and this generated strange behavior. Examples included Jude mistakenly firing at the player while standing right in front of them and Jude walking directly up to the player and then freezing. The firing appears to be a glitch, but the behavior of walking directly up to a stationary player and stopping right in their face was intentional, though simplistic. When a player is moving, this behavior maintains a fair distance between the player and Jude.

Oddly enough, three players suggested that Jude could have stuck closer to them. If the player was not often in Jude's field of vision, the "following" behavior would not activate or Jude would lose track of the player. Also, unlike a human player who can predict a teammate's direction based on their movement, Jude follows teammates by tracking their last known location, but cannot predict a player's movement from that. This means that Jude could be prone to lagging behind or losing the player entirely if it couldn't see them again. Despite this limitation, most users felt that Jude was a good follower.

Some participants thought that Ethan was following them, but this would be an incidental result of Ethan running in the same direction as the player, as Ethan has no explicit instructions to follow teammates.

Participants said multiple times that a bot would lead them to enemies or see enemies before the player did. Players assumed that the bots could see enemies through the scenery. One remarked that they saw Jude "shooting at a door before the enemy bots would enter" and explained that they assumed Jude already knew where the enemies were, while another suggested we improve the bot by having it "act like it didn't know where the enemies are." These subjects were incorrect in their assumption that the bots had unfair awareness of enemy locations. Pogamut only provides information about what the bots can see, although the bots will detect the presence of agents with more speed and fidelity than a typical human. They can identify enemies even from brief glimpses through a closing/opening door for example. The player's misconception likely originated from past experience with games where computer controlled agents have a greater awareness of the map than the ones in this experiment.

V. DISCUSSION AND FUTURE WORK

Users expressed clear preferences for helpful bots that they see frequently, and there was also some indication that players like teammates that follow them, as well as score well and avoid death well. However, being too good is definitely problematic, as indicated by user comments and the fact that these preferences are not significant. Also, following too much can be annoying, especially if it means a bot is constantly in the player's personal space. This study focused on extreme behaviors to create a clear distinction between the two bots (extreme focus on following vs. extreme focus on performance by scoring well and avoiding death), but players would likely prefer a middle ground combining the best behaviors of both bots. A bot that follows well, but not too close, and is good at scoring and avoiding death, but not significantly better than the player, would be both effective and enjoyable to play with.

Though users preferred helpful bots, the term "helpful" seems to have been too vague, and in retrospect should have been replaced with more specific terms. Without a concrete definition of what "helpful" meant, players were free to interpret this question in a variety of ways. The intent of this question was to measure how cooperative and team-oriented each bot was, but many participants seem to have interpreted behavior leading to a higher score as more helpful. However, some users, in particular the two that gave Ethan a helpfulness rating of 1, seem to have interpreted this question differently. It is unclear what bot behaviors these players were interpreting as helpful. Any future studies similar to this one would need to clarify this issue in the survey questions. Questions could also be added to gauge how much a player's preference is influenced by the final score compared to the bot's behavior during the round.

Multiple modern games create team-oriented companions by providing companions who have to be directed through a fight and will die without player intervention. Though some players are annoyed by this, games in the "tactical shooter" genre have been successful with players who enjoy a feeling of collaboration. Even with a fully autonomous companion, many players like to feel that they are moving and working with their companion which can be achieved by having the companion stay closer to the player even when it may be more advantageous for the two to split up.

Several participants commented that the bots jumped too often. However, unnecessary jumping is also characteristic in human players across multiple games. In a 2015 youtube video, the popular gaming channel *outsidexbox* listed jumping everywhere as one of the "9 things you can't resist doing in videogames." In the video, they explain that if jumping is just as fast as running, they would "bunyhop all the way [to my destination] instead of running like a sensible hero" [22]. Given that this tendency for jumping is common to human players, testers may have noticed it more in a computer controlled teammate because they did not expect this kind of human behavior to be replicated. This behavior might simply be more visible when the player is not the one doing it.

Hopefully, this paper's findings can inform the development of companion characters in future games. This study found that there are players that prioritize experience and the feeling of teamwork over the number of points they gain, which could be helpful in games where story is the focus rather than score. Of course, industry is already aware of this issue, which is part of the reason that state-of-the-art AI techniques for agent control are seldom adopted in commercial games. Ethan demonstrates how an agent controlled by a sophisticated AI technique (neuroevolution) can perform a task well (scoring points), but in a way that a human player may find frustrating. Still, despite some negative comments, players did slightly prefer Ethan over Jude, so perhaps neuroevolution (and other techniques) are not so impractical as mechanisms for controlling bots. In fact, one could replace Jude's simplistic combat behavior with Ethan's evolved combat behavior, but keep Jude's scripted behaviors for following, and get the best of both. In fact, UT^2 [7] is a previous example of a bot that combines complicated scripting with evolved behavioral modules. Perhaps agents in commercial games can do the same.

Future studies could examine other game variants that require more teamwork. Team deathmatch can be won simply by earning the most kills, even if there is no teamwork. Other game variants such as Capture the Flag, Domination, and Assault tend to require more teamwork, and benefit greatly from communication between teammates about what goals to focus on. The bots in this study were incapable of communication, and it is unclear how cooperative such bots can be without this capability. The popularity of these more complicated team modes of play also makes them of greater interest to industry, and thus deserving of future focus.

VI. CONCLUSION

This study shows that though players may not show a clear preference for one bot over the other, the reasons behind their preferences still follow noticeable trends. Player preference was found to depend most strongly on which bot they saw more often and which one they found to be more helpful. Better following can lead to being seen more, though this is also a matter of chance. Being helpful can be interpreted multiple ways, such as being good at winning, or as being more collaborative and team-focused. Bots that combine these behaviors will presumably be more enjoyable for humans to play with. This hypothesis will be further evaluated in the future, in more challenging variants of team play.

ACKNOWLEDGMENT

This research is supported in part by the Summer Collaborative Opportunities and Experiences (SCOPE) program, funded by various donors to Southwestern University.

REFERENCES

[1] P. Hingston, *Believable Bots: Can Computers Play Like People?* Springer, 2012.

[2] P. Hingston, "A Turing Test for Computer Game Bots," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 1, no. 3, pp. 169–186, 2009.

[3] P. Hingston, "A New Design for a Turing Test for Bots," in *Computational Intelligence and Games*. IEEE, 2010.

[4] M. Polceanu, "MirrorBot: Using Human-inspired Mirroring Behavior to Pass a Turing Test," in *Computational Intelligence and Games*, 2013.

[5] J. Schrum, I. V. Karpov, and R. Miikkulainen, "Humanlike Combat Behavior via Multiobjective Neuroevolution," in *Believable Bots*. Springer, 2012, pp. 119–150.

[6] I. V. Karpov, J. Schrum, and R. Miikkulainen, "Believable Bot Navigation via Playback of Human Traces," in *Believable Bots*. Springer, 2012, pp. 151–170.

[7] J. Schrum, I. V. Karpov, and R. Miikkulainen, "UT^2: Human-like Behavior via Neuroevolution of Combat Behavior and Replay of Human Traces," in *Computational Intelligence and Games*, 2011.

[8] K. O. Stanley and R. Miikkulainen, "Evolving Neural Networks Through Augmenting Topologies," *Evolutionary Computation*, vol. 10, pp. 99–127, 2002.

[9] J. Schrum and R. Miikkulainen, "Discovering Multimodal Behavior in Ms. Pac-Man through Evolution of Modular Neural Networks," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 8, no. 1, pp. 67–81, 2016.

[10] K. E. Merrick and M. L. Maher, *Motivated Reinforcement Learning: Curious Characters for Multiuser Games*. Springer, 2009.

[11] C. Guckelsberger, C. Salge, and S. Colton, "Intrinsically Motivated General Companion NPCs via Coupled Empowerment Maximisation," in *Computational Intelligence and Games*, 2016.

[12] A. T. Abraham and K. McGee, "AI for Dynamic Team-mate Adaptation in Games," in *Computational Intelligence and Games*, 2010, pp. 419–426.

[13] K. McGee and A. T. Abraham, "Real-time Team-mate AI in Games: A Definition, Survey, & Critique," in *Foundations of Digital Games*. ACM, 2010, pp. 124–131.

[14] J. E. Laird and M. v. Lent, "Human-Level AI's Killer Application: Interactive Computer Games," in *National Conference on Artificial Intelligence*. AAAI Press, 2000, pp. 1171–1178.

[15] A. M. Turing, "Computing Machinery and Intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[16] J. Gemrot, R. Kadlec, M. Bida, O. Burkert, R. Pibil, J. Havlicek, L. Zemcak, J. Simlovic, R. Vansa, M. Stolba, T. Plch, and B. C., "Pogamut 3 Can Assist Developers in Building AI (Not Only) for Their Videogame Agents," *Agents for Games and Simulations, LNCS 5920*, pp. 1–15, 2009.

[17] N. Kohl and R. Miikkulainen, "An Integrated Neuroevolutionary Approach to Reactive Control and High-level Strategy," *IEEE Transactions on Evolutionary Computation*, 2011.

[18] K. O. Stanley, B. D. Bryant, and R. Miikkulainen, "Evolving Neural Network Agents in the NERO Video Game," in *Computational Intelligence and Games*, 2005.

[19] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 182–197, 2002.

[20] N. van Hoorn, J. Togelius, and J. Schmidhuber, "Hierarchical Controller Learning in a First-Person Shooter," in *Computational Intelligence and Games*. IEEE, 2009, pp. 294–301.

[21] A. Marx, C. Backes, E. Meese, H.-P. Lenhof, and A. Keller, "EDISON-WMW: Exact Dynamic Programing Solution of the Wilcoxon-Mann-Whitney Test," *Genomics, Proteomics & Bioinformatics*, vol. 14, no. 1, pp. 55–61, 2016.

[22] M. Channell, J. Douglas, and A. Farrant, "9 things you can't resist doing in videogames," https://www.youtube.com/watch?v=uTEQESZS3Eo, May 2015.