Passing a Hide-and-Seek Third-Person Turing Test

Andrew Cenkner, Vadim Bulitko, Marcia Spetch, Eric Legge, Craig G. Anderson, and Matthew Brown

***Abstract*—Hiding and seeking are cognitive abilities frequently demonstrated by humans in both real life and video games. To determine to which extent these abilities can be replicated with AI, we introduce a specialized version of the Turing test for hiding and seeking. We then develop a computer agent that passes the test by appearing indistinguishable from human behavior to a panel of human judges. We analyze the AI techniques that enable the agent to imitate human hide-and-seek behavior and their relative contribution to the agent's performance.**

***Index Terms*—AI bots, hide-and-seek behavior, Turing test.**

I. Introduction

HIDING and seeking are considered to be nontrivial human cognitive behaviors that develop with age [1], [2]. These behaviors are present in a variety of video games in some form. For instance, competitive online first-person shooters such as *Counter-Strike: Source* [3] have players searching for members of the opposing team (e.g., snipers). Role-playing games such as *Borderlands* [4] or *Fallout: New Vegas* [5] encourage the player to explore the environment and reward such exploration with weapons, side quests, and information on the story and the environment.

To support these hide-and-seek activities, game developers face several challenges. First, level designers need to place desirable items ("loot") in locations that would reward both casual and hardcore players. Deciding on which kinds of items to place at which locations can be made easier and more efficient by predicting, at the game development stage, where the players will search and how their search patterns will depend on the player type (e.g., from a casual player to a completionist).

Second, game designers can enhance behavior of in-game AI-controlled agents with knowledge of where the players will be looking for other players (e.g., in *Counter-Strike: Source*) or other players' units (e.g., in *StarCraft 2* [6]). This would allow the AI agents to better control the level of stealth they exhibit in the game.

Manuscript received December 04, 2012; revised March 26, 2013; accepted June 28, 2013. Date of publication July 30, 2013; date of current version March 13, 2014. This work was supported by the National Science and Engineering Research Council.

A. Cenkner and V. Bulitko are with the Department of Computing Science, University of Alberta, Edmonton, AB T6G 2E8 Canada (e-mail: cenkner@ualberta.ca; bulitko@ualberta.ca).

M. Spetch, E. Legge, and C. G. Anderson are with the Department of Psychology, University of Alberta, Edmonton, AB T6G 2E8 Canada (e-mail: mspetch@ualberta.ca; elegge@ualberta.ca; cganders@ualberta.ca).

M. Brown is with the Psychiatry Department, University of Alberta, Edmonton, AB T6G 2E8 Canada (e-mail: mbrown2@ualberta.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCIAIG.2013.2275162

Finally, game developers need to develop nonplayable characters that search for the player in a compelling way. A common approach is to give such characters a perfect knowledge of the player's position and then add hard-coded behavior obfuscating such omniscience. Natural-looking obfuscation is labor intensive as it requires extensive trial-and-error iterations and may be fragile insomuch as every once in a while the characters demonstrate their omniscient knowledge of player's position. This is viewed as "cheating" in video games and can break player's immersion.

Beyond video games, understanding hiding and seeking is valuable to law-enforcement agencies (e.g., predicting hiding spots for illegal substances) and the military (e.g., predicting locations of weapon stashes, improvised explosive devices, enemy troops, or sniper positions). From a theoretical and cognitive perspective, if hiding and seeking are fundamental cognitive abilities of humans, then understanding them via a computer program/model may shed light on human cognition and/or bring us closer to building strong AI.

The rest of the paper is organized as follows. We formalize the problem and describe our performance measures in Section II. In Section III, we review the existing work and argue that it is insufficient to solve the problem at hand. Our own approach is presented in Section IV, followed by an empirical evaluation. We then discuss the results, consider directions for future work (Section VII) and conclude the paper.

This paper extends our conference publication [7] by offering more details on the approach, an extensive walkthrough example, two new AI agent designs, and new empirical results.

II. Problem Formulation

The problem we are tackling in this paper is to develop an AI agent that hides and seeks objects in a virtual environment. The criteria for success will be whether the agent does so in a humanlike fashion, replicating peculiarities of human hiding and seeking behavior. How similar the agent's behavior is to that of humans is to be judged by a panel of humans. Note that the AI agent is tested in a novel environment where it has not previously seen any humans hiding or seeking.

Tests in which humans judge how humanlike a computer is are known as Turing tests, after the original test proposed by Alan Turing [8]. In game-like settings, such tests are also known as player believability tests, since, to pass such a test, a computer agent controlling the player's avatar needs to mislead the judges to believe that the player behind the avatar is human [9]. Because of the difficulty of passing the full Turing test, specialized or restricted Turing tests are commonly used. The Loebner competition [10] and the BotPrize [11], [12] use restricted Turing tests to assess player believability of computer programs.

Early Turing tests assume that the task the agent is being judged on allows the agent to interact with the human judge (e.g., via a conversation in the Loebner competition or in first-person multiplayer shooter settings in the BotPrize). Later Turing tests have been generalized to single-player tasks where the judge is necessarily a passive observer of the agent. Some researchers have even argued that such third-person tests are actually more accurate than first-person tests as the judge is able to separate their judging from their game-play and, therefore, "is able to concentrate more on the assessment of believability via a higher cognitive focus on the task" [9]. A recent example of such a third-person Turing test is the Turing test track of the international Mario AI Championship [13].

Our specialized version of the Turing test is to have an AI agent exhibit hide-and-seek behavior in a novel environment that is indistinguishable (by a panel of human judges) from humans operating in the same environment. This hide-and-seek task we are interested in is inherently single player, and, thus, our Turing test will use third-person assessment of player believability by having the human judges passively observe agent behavior.

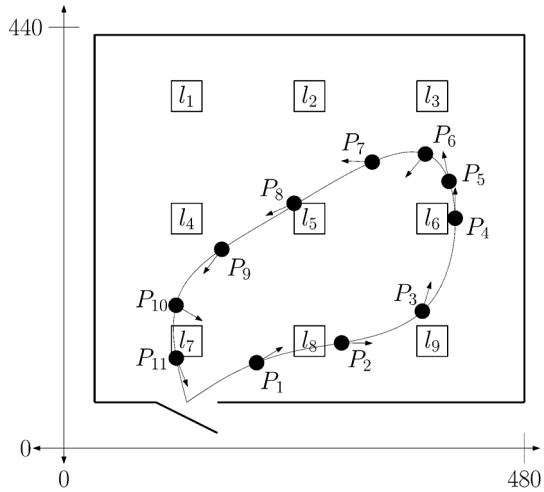
A. Telemetry

We formalize our test as follows. Two different environments are prepared. Each environment has a finite set $L = \{l_1, \ldots, l_n\}$ of n discrete locations where objects can be hidden or sought. Examples of such locations are a desk drawer, a window sill, a floor tile, or a discolored area of paint on a wall. In our experiments, the locations were floor tiles. The participants (humans and AI agents) are tasked with repeatedly hiding or seeking objects in both environments. They do so by moving about the environment and occasionally hiding an object at one of the locations or checking a location for a hidden object.

We define a *route* to be a recording of one participant completing either the hide or seek task. The route is recorded as their path P and location–selection history S. The path $P = (P_1, \ldots, P_m)$ is a list of the participant's Cartesian coordinates and orientations at different points in time. Similarly, the selection history $S = (S_1, \ldots, S_p)$ is a list of the participant's location selections and the corresponding times. Mathematically, $P_i = (x_i, y_i, \phi_i, t_{P_i})$, where $1 \leq i \leq m$; $x_i, y_i, t_{P_i} \in \mathbb{R}$; and $\phi_i \in [0^\circ, 360^\circ)$; $S_i = (l_{S_i}, t_i)$, where $1 \leq i \leq p$; $l_{S_i} \in L$; and $t_i \in \mathbb{R}$.

To illustrate, Fig. 1 shows a hypothetical sample of one participant completing the seek task in a simple room, with $m = 11$ and $p = 6$. The participant enters the room at the door on the bottom left. He/she then travels around the room along the path shown in the figure. The participant's location is recorded every second as $P_i, 1 \leq i \leq 11$ and shown in the figure as filled circles. The participant also seeks at the six locations $S_j, 1 \leq j \leq 6$: at the location l_8 at time 1.1, at the location l_9 at time 2.6 and so on as listed in the figure.

When combining several routes together, we will use superindices to group locations by route. For instance, suppose in addition to the route shown in Fig. 1, we have another route consisting of a single location access: $(l_1, 4.4)$. Then, we can represent both routes as $S = (S^1, S^2)$ where the first route S^1 consists of six locations: $S^1 = (S^1_1, S^1_2, S^1_3, S^1_4, S^1_5, S^1_6) = ((l_8, 1.1), (l_9, 2.6), (l_6, 3.2), (l_3, 5.1), (l_5, 7.7), (l_7, 9.8))$, while the second route has one: $S^2 = (S^2_1) = ((l_1, 4.4))$. Also, when the specific time points at which the locations were accessed are not important, we will use a simplified notation. The aforementioned routes S^1 and S^2 can be timelessly represented as Q^1 and Q^2: $Q^1 = (Q^1_1, Q^1_2, Q^1_3, Q^1_4, Q^1_5, Q^1_6) = (l_8, l_9, l_6, l_3, l_5, l_7)$ and $Q^2 = (Q^2_1) = (l_1)$.



$P_1 = (201, 89, 31^\circ, 1.0)$ $P_2 = (289, 110, 0^\circ, 2.0)$
$P_3 = (374, 143, 69^\circ, 3.0)$ $P_4 = (408, 240, 90^\circ, 4.0)$
$P_5 = (401, 279, 97^\circ, 5.0)$ $P_6 = (377, 308, 225^\circ, 6.0)$
$P_7 = (321, 299, 179^\circ, 7.0)$ $P_8 = (240, 256, 201^\circ, 8.0)$
$P_9 = (165, 208, 227^\circ, 9.0)$ $P_{10} = (117, 149, 329^\circ, 10.0)$
$P_{11} = (117, 94, 292^\circ, 11.0)$
$S_1 = (l_8, 1.1)$ $S_2 = (l_9, 2.6)$ $S_3 = (l_6, 3.2)$
$S_4 = (l_3, 5.1)$ $S_5 = (l_5, 7.7)$ $S_6 = (l_7, 9.8)$

Fig. 1. Top–down view of an environment with nine possible locations shown as squares. The participant performed the seek task, and his/her path is shown. The path is sampled at 11 moments of time $(P_1, \ldots, P_{11})$. The six locations the participant sought at are recorded as $(S_1, \ldots, S_6)$.

B. The Test

Human routes recorded from one of the two environments (called the *training environment*) constitute the *training data* and are made available to the AI agent. The AI agent is then asked to create its own routes in the other environment (called the *test environment*) and its routes are recorded. A group of human judges are then individually presented with a playback of the routes created by humans and the AI agent in the test environment. Each judge is asked to label each route as either "human" or "computer." The entire process is then repeated with the two environments swapped in their training/test roles. We define an agent's *identification rate* as the percentage of the agent's routes the judges correctly label. An agent passes our Turing test if the 95% confidence interval for its identification rate falls between the identification rates of 45% and 55%. That is to say, an agent passes our test if we can claim with 95% confidence that the identification rate falls within 5% of the chance level. Note that the human judges are trained at this task by being exposed to human routes prior to the test.

Our objective of minimizing the difference between the agent's identification rate and the chance level rate raises some important questions. What if the judges are unmotivated

and randomly guess at all times? What if the routes are not presented in an intelligible way to the judges? Is the task at which the agent is being tested cognitively rich so that passing the test is a meaningful achievement in believable agent design or understanding intelligence?

We will first argue that the specialized version of the Turing test is cognitively rich. Research in psychology has demonstrated that hiding and seeking appear to involve skills in orientation, navigation, and the theory of mind [1]. These skills are nontrivial and humans hone these skills as they mature [2]. Other animals vary greatly in their performance in hiding and seeking with species having more highly evolved spatial abilities exhibiting better performance [14].

The ecological validity of virtual environments as tests of navigation and memory has been suggested through experiments that have shown transfer of spatial knowledge from virtual environments to the real space counterpart in both adults [15] and children [16]–[18]. Although differences in performance between virtual and real environments are sometimes observed (e.g., a greater tendency to underestimate distance in virtual environments [19]), many fundamental findings hold in both environments. For instance, researchers found that in both virtual and real environments Alzheimer disease patients were impaired relative to controls in navigating to a goal when cues on the walls provided the only cues for navigation but not when start position provided the only cues for navigation [20]. Others found that the developmental process of spatial knowledge acquisition was comparable in real and virtual maze tasks [21]. Moreover, functional magnetic resonance imaging (fMRI) studies in humans [22] showed distinct neural activation for wayfinding and route following in a virtual town, which corresponds to the neural dissociations shown in rodents for place and response learning in real maze tasks. Consequently, virtual environments have been used extensively over the past couple of decades to investigate neural and behavioral processes underlying spatial navigation [22]–[24] and to investigate, diagnose, or rehabilitate spatial cognitive impairments [18], [20], [25]–[27], and even to develop strategies to interdict terrorists carrying radioactive material [28].

Directly relevant to the present work, we have shown in two previous studies [1], [29] that hiding and searching behavior in virtual rooms is similar in many respects to that seen in real-life rooms. In these experiments, people searched for and hid objects in bins or under tiles on the floor of a room. In our first set of studies [1], the room was a simple rectangular space with nine bins that provided hiding and searching locations. Importantly, the same systematic differences between hiding and searching emerged in both real and virtual versions of the room. Specifically, in both environments, people chose locations farther from origin, and they dispersed their choices more when hiding than when searching. In our second set of studies [29], people were tested in more complex virtual and real rooms that contained several dozen hiding locations provided by tiles on the floor of the room. In some conditions, the rooms were empty and, in others, they were cluttered with furniture. For our purposes, the main finding of interest was that, in both empty and cluttered rooms, people continued to choose locations farther from origin and to disperse their choices more when hiding than when searching. The generality of these differences between hiding and searching behavior across real and virtual tasks and across simple empty spaces and more cluttered complex spaces justified the use of an empty virtual room for the purpose of our study.

To argue that the information given to the judges is sufficiently rich and that the judges are properly motivated and skilled, we develop several reasonably complex hide-and-seek agents and show that they are easily distinguished from humans by the judges. Admittedly, being "reasonably complex" is a matter of opinion and we, as the authors, may be biased. So we present the "reasonably complex" designs in the paper for the reader to judge.

Finally, we will briefly investigate whether human judges become better with practice and whether they appear to distribute labels evenly between "human" and "computer."

III. Related Work

A. Related Work in Psychology

Work in psychology indicates that hiding and seeking strategies are cognitive processes. Birds of higher spatial and social intelligence develop more complex hide-and-seek strategies related to their survival, especially with respect to food caching [30]–[32]. Also, the complexity of children's hide-and-seek strategies varies with age [33], [34].

More recent studies [1], [29] considered hiding and seeking behavior of adult humans in a virtual environment. The observed behavior was analyzed only at an aggregate level (e.g., mean distance traveled from the room entrance to the first hiding location). The published results focus on correlations between hide or seek frequency at certain locations and room features. For example, Legge *et al.* [29] show that there is a positive correlation between a location's distance from walls and hiding frequency at such locations, but a negative correlation between distance from walls and seeking frequency. This indicates that humans do not seek in the same places they hide. These kinds of observations are helpful when mapping out the basic framework of an agent, but are not sufficiently specific to predict particular hide-and-seek locations. To the best of our knowledge, no generative computational models useful for an AI agent mimicking hide-and-seek behavior have been published in psychology literature to date.

B. Related Work in Computing Science

The Turing test, originally called the "imitation game," was introduced in 1950 [35] and is often linked to believability of AI agents [36]. There are two well-known public competitions related to the Turing test. The Loebner Prize [10] test is the classic teletype version of the Turing test and has not been passed to date. The BotPrize [12] replaces the teletype environment with a competitive first-person shooter environment. To compete, an AI agent is no longer required to communicate, but must move and engage in combat in a way indistinguishable from humans (as deemed by human judges). There have been a number of unsuccessful attempts to pass the test (e.g., [37]–[39]) and only very recently the first positive results were reported [40]. While the technical details are still scarce, it is unclear how to apply

the winning designs to our problem of humanlike behavior in the tasks of hiding and seeking objects in a novel environment. There is no combat and no opponents—both critical components of the BotPrize test.

C. Related Work in Video Games

Historically, in first-person shooters, an AI agent is created by selecting behaviors believed to be important for the agent to appear human and hard-coding them into the game. For example, in the game *Counter-Strike: Source*, the agents were programmed to specifically mimic the slow reaction time and attention prioritization displayed by humans [41]. Other researchers [42] used visibility maps and Boolean logic to make nonplayable characters (NPCs) take cover safely. These types of hand-coded approaches are game/task-specific and may not be portable to other tasks.

There has been work on predicting possible player locations in first-person shooter games. Some proposed a model using particle filters to predict human locations, removing the dependence on cheating with omniscient knowledge [43]. The particle filter approximates opponent locations by simulating possible opponent paths from the last known location. Others have improved on the particle filter method by adding hidden Markov models and simulacra [44], [45]. These prediction models are promising but they do not demonstrate how to actually create an autonomous agent. Additionally, these models do not offer portability as they require training data for the very environment they are used in.

Similarly to the human prediction models for first-person shooter games, there has been work in creating human prediction models for real-time strategy (RTS) games. Researchers have demonstrated how to predict multiple units' paths given limited observations, using an assumption that units only make small deviations from optimal paths [46], [47]. Although these approaches are helpful to an RTS game designer, the optimal path assumption may not be humanlike. More recently, researchers tracked units after losing sight using a particle model [48]. The model shows promising predictions, but similarly to its first-person shooter particle counterparts, it provides no way to collapse the predictions into a single humanlike course of action that would be needed to pass our version of the Turing test.

IV. Proposed Approach

In order to pass the specialized Turing test described in Section II, we needed a model capable of generating humanlike hide-and-seek routes. The model was to be trained on human routes recorded in one environment and then applied to generate new routes in another, previously unseen, environment. We also wanted to have variability in our model so that it generates different routes over multiple runs in the same environment. Finally, we set out to automatically learn as much of the model as possible using the training data, thereby reducing the manually engineered component.

We decomposed route generation into two sub-behaviors, selecting locations and navigating among them. The decomposition made the design more modular as well as gave us insight into the relative contributions of each sub-behavior. For location–selection sub-behavior, we created a simple strategy (L1) and an advanced strategy (L2). Likewise, for the movement sub-behavior we developed a simple strategy (M1) and an advanced strategy (M2). In order to explain each strategy in detail, we include a small artificial example running throughout this section.

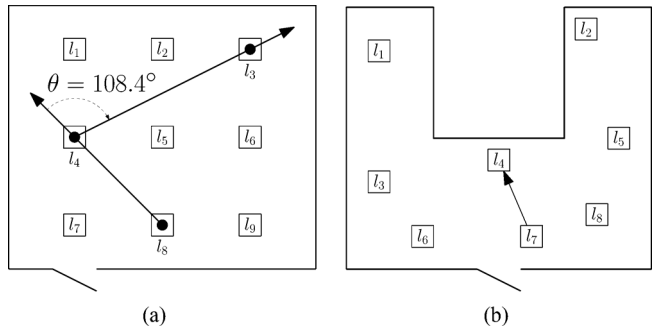


Fig. 2. Training and testing rooms used in our running example. (a) Example of $G(l_8, l_4, l_3)$. (b) Locations in the testing room.

The training and testing environments for our running example are in Fig. 2(a) and (b), respectively. In the example, an agent starts at the door in the testing room which contains eight floor tiles where the participant can hide/seek objects. The agent entered the room, sought at the location l_7, then at the location l_4, and is now looking for the next location to select. We will now demonstrate how the next location is selected under the L1 and L2 strategies.

A. Location–Selection Strategies

Each of the two selection strategies (L1 and L2) is given an ordered list of locations already selected by the agent, and asked to make a new selection. Given the list, the strategies assign probabilities to each location in the room, thereby imposing a probability distribution. The next selection is then stochastically drawn from the distribution, with replacement for the seek task and without replacement for the hide task to reflect the location revisitation practices of humans.

Let P_{L1} and P_{L2} represent the probability distribution functions (pdfs) for strategies L1 and L2, respectively. Every location receives a probability between 0 and 1, and the sum of all the locations' probabilities is 1.

1) Strategy L1 (Uniform Random Selection): The strategy assigns equal probability of selection to all eligible locations.

2) Strategy L2 (Data-Driven Location Selection): The strategy is based on three probability distributions: $P_{\mathcal{D}}$, $P_{\mathcal{A}}$, and $P_{\mathcal{R}}$, created automatically from the training data (i.e., human location selections observed in the training environment). The distribution $P_{\mathcal{D}}$ is based on the distance between consecutively selected locations, $P_{\mathcal{A}}$ is based on the rotation angle between consecutively selected locations, and $P_{\mathcal{R}}$ is based on the last time the location was selected. Then, L2 uses these three probability distributions each time it is asked to select the next location. Specifically, each location receives three probability values, one for each distribution. A product of these three distributions comprises L2's final distribution: $P_{\mathrm{L2}} = P_{\mathcal{D}} \times P_{\mathcal{A}} \times P_{\mathcal{R}}$. We will now detail each of the three distributions.

	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
l_1	0	1	2	1	1.41	2.24	2	2.24	2.83
l_2	1	0	1	1.41	1	1.41	2.24	2	2.24
l_3	2	1	0	2.24	1.41	1	2.83	2.24	2
l_4	1	1.41	2.24	0	1	2	1	1.41	2.24
l_5	1.41	1	1.41	1	0	1	1.41	1	1.41
l_6	2.24	1.41	1	2	1	0	2.24	1.41	1
l_7	2	2.24	2.83	1	1.41	2.24	0	1	2
l_8	2.24	2	2.24	1.41	1	1.41	1	0	1
l_9	2.83	2.24	2	2.24	1.41	1	2	1	0

(a)

	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
l_1	1	2	4	2	3	5	4	5	6
l_2	2	1	2	3	2	3	5	4	5
l_3	4	2	1	5	3	2	6	5	4
l_4	2	3	5	1	2	4	2	3	5
l_5	3	2	3	2	1	2	3	2	3
l_6	5	3	2	4	2	1	5	3	2
l_7	4	5	6	2	3	5	1	2	4
l_8	5	4	5	3	2	3	2	1	2
l_9	6	5	4	5	3	2	4	2	1

(b)

Fig. 3. (a) Pairwise location distances and (b) their ranks.

a) Spatial distance pdf: The pdf $P_{\mathcal{D}}$ for spatial distance between consecutively selected locations is computed as follows. First, we rank all possible distances between location pairs in the training room. Then, we assign a weight to each distance as the ratio of the number of times the distance occurred between consecutive human selections to the number of times the distance occurs between all location pairs in the room.

Mathematically, we first build a set of all possible unique location pair distances: $D = \{E(l_i, l_j) \mid 1 \leq i, j \leq n\}$ where $E(l_i, l_j)$ is the Euclidean distance between the centers of locations l_i and l_j. We sort the set D in the ascending order and build an index function $I_D : D \rightarrow \{1, \ldots, |D|\}$ such that, for any possible distance d between two locations, $I_D(d)$ gives d's index in the sorted set D. Thus, $I_D(E(l_i, l_j))$ represents the rank of the distance between the location l_i and l_j among all distances between location pairs.

For example, in our training environment, there are six unique distances between all pairs of locations. We have $I_D(E(l_1, l_1)) = I_D(0) = 1$ (the shortest distance is zero), $I_D(E(l_5, l_6)) = 2$ (the distance between locations 5 and 6 is tied for the second shortest distance with many other location pairs), and $I_D(E(l_1, l_9)) = 6$ (the distance between locations 1 and 9 is tied for the largest distance). Fig. 3 lists $E(\cdot)$ and $I_D(E(\cdot))$ for the training environment.

First, we initialize the pdf to zero: $P_{\mathcal{D}}(i) = 0, 1 \leq i \leq |D|$. Next, we consider each selection history $Q^j = (q_1^j, \ldots, q_{n_j}^j)$ in the training data. For each pair of consecutive locations in Q^j, we compute the Euclidean distance $d = E(q_i^j, q_{i+1}^j)$. Then, we increase the frequency of the corresponding index $I_D(d)$ in the pdf. The increase is scaled by dividing by the number of times that distance occurs in the environment. Formally, if X represents the number of location pairs in the environment whose distance is d

$$X = |\{(l_a, l_b) \mid E(l_a, l_b) = d, 1 \leq a, b \leq n\}| \tag{1}$$

then we update $P_{\mathcal{D}}$ as

$$P_{\mathcal{D}}(I_D(d)) \leftarrow P_{\mathcal{D}}(I_D(d)) + \frac{1}{X}. \tag{2}$$

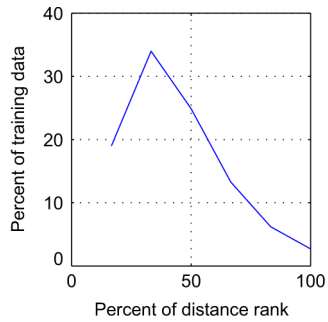
To illustrate, the location pair counts in the training data are shown in Fig. 4. For instance, the location l_1 was selected immediately after the location l_9 nine times. The distance between these two locations is $d = E(l_1, l_9) = 2.83$ as per Fig. 3.

TABLE I
SPATIAL DISTANCE PDF FOR OUR EXAMPLE

Rank : $1 \ldots \lvert D\rvert$	1	2	3	4	5	6
Number selected	$\frac{528}{9}$	$\frac{2512}{24}$	$\frac{1227}{16}$	$\frac{492}{12}$	$\frac{305}{16}$	$\frac{33}{4}$
$P_{\mathcal{D}}(i)$	0.190	0.340	0.249	0.133	0.062	0.027

	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
l_1	60	101	40	98	80	22	45	27	6
l_2	109	52	90	82	111	76	13	39	21
l_3	35	110	67	19	73	101	11	23	32
l_4	93	89	14	43	82	41	123	72	26
l_5	69	125	62	76	48	112	68	109	73
l_6	25	77	99	40	107	72	15	76	100
l_7	38	34	7	92	83	12	76	134	76
l_8	18	43	9	80	88	81	89	62	121
l_9	9	19	35	8	86	120	28	122	48

(a)



(b)

Fig. 4. (a) Location pair selection counts from the training data. (b) Spatial distance pdf $P_{\mathcal{D}}$ built from it.

The distance 2.83 has a rank of 6 in this environment (Fig. 3) and will thus be contributing to $P_{\mathcal{D}}(I_D(2.83)) = P_{\mathcal{D}}(6)$. Three other location pairs contribute to the value of the pdf for rank 6: $(l_7, l_3), (l_3, l_7), (l_1, l_9)$, which collectively happen in the training data $7 + 11 + 6$ times. Thus, after the training data are processed, the pdf value for rank 6 is set to $P_{\mathcal{D}}(6) = (9 + 7 + 11 + 6)/4 = 33/4$ since $X = 4$ (there are four location pairs with the distance of 2.83). The ratio is found in the last column of Table I. Once all training data are processed, the $P_{\mathcal{D}}$ values are normalized

$$P_{\mathcal{D}}(i) \leftarrow \frac{P_{\mathcal{D}}(i)}{\sum_{j=1}^{|D|} P_{\mathcal{D}}(j)}, \qquad 1 \leq i \leq |D|. \tag{3}$$

The resulting values are listed in the bottom row of Table I as well as displayed in Fig. 4.[1]

b) Rotation angle pdf: To compute $P_{\mathcal{A}}$, we start by considering all triplets of locations. For any three locations l_a, l_b, and l_c, we define $G(l_a, l_b, l_c)$ as the angle between $\overrightarrow{l_a l_b}$ and $\overrightarrow{l_b l_c}$.[2] An example of $G(l_8, l_4, l_3) = 108.4°$ is shown in Fig. 2(a). We form the set of all unique angles $A = \{G(l_a, l_b, l_c) | l_a \neq l_b \neq l_c, 1 \leq a, b, c \leq n\}$, sort the set in an ascending order, and build an index function $I_A : A \rightarrow \{1, \ldots, |A|\}$ such that for any angle $\theta \in A$, $I_A(\theta)$ gives its index in the sorted set A.

Then, we process the training data. Once again, we consider each selection history $Q^j = (q_1^j, \ldots, q_{n_j}^j)$. Next, for each location triplet $(q_{i-1}^j, q_i^j, q_{i+1}^j)$ in it, we compute the angle $\theta = G(q_{i-1}^j, q_i^j, q_{i+1}^j)$ and update the frequency of the corresponding index $I_A(\theta)$ in the pdf. The update is scaled by the number of times that angle occurs in the environment. Formally,

[1]This artificial example is given for illustration purposes only. The data in Fig. 4 are made up. The pdf for the actual data is shown in Fig. 8.

[2]We ignore all cases where l_a, l_b, l_c line up in a way that makes the angle $G(l_a, l_b, l_c)$ undefined (e.g., $l_b = l_c$).

if X represents the number of angles in the environment with the same rank as θ

$$X = |\{(l_a, l_b, l_c) \mid I_A(G(l_a, l_b, l_c)) = I_A(\theta), 1 \le a, b, c \le n\}| \tag{4}$$

then we update $P_{\mathcal{A}}$ as

$$P_{\mathcal{A}}(I_A(\theta)) \leftarrow P_{\mathcal{A}}(I_A(\theta)) + \frac{1}{X}. \tag{5}$$

Finally, we normalize the frequency for each rank to get the pdf over the angle ranks

$$P_{\mathcal{A}}(i) \leftarrow \frac{P_{\mathcal{A}}(i)}{\sum_{j=1}^{|A|} P_{\mathcal{A}}(j)}, \qquad 1 \le i \le |A|. \tag{6}$$

c) Selection recency pdfs: For the last distribution, we will first define a *recency number*. Given a sequence of locations the agent has already hid/sought at, $Q = (q_1, \ldots, q_n)$, the recency number of the location l is the number of locations since and including the most recent access to l in Q. Mathematically, $R(l|Q) = |Q| - \max\{i \mid q_i = l\} + 1$. If the location l has not been previously accessed ($l \notin Q$), then $R(l|Q) = \infty$. To illustrate, suppose a location–selection history is $Q = (l_8, l_9, l_6, l_3, l_5, l_7)$ as per Fig. 1. Then, $R(l_7|Q) = 1$, $R(l_2|Q) = \infty$, and $R(l_6|Q) = 4$.

Suppose the training data consist of J location sequences $Q^j = (q_1^j, \ldots, q_k^j, \ldots)$ where $j, 1 \le j \le J$ is the sequence (i.e., route) number. Then, each location q_i^j has a recency number with respect to the part of its route ending immediately before q_i^j. Considering the entire training data set Q, such recency numbers run between 1 and ∞. With a finite data set, there is a highest finite recency number, denoted by $r_{\max}$.

To illustrate, let us suppose the training data consist of two routes $Q^1 = (l_8, l_9, l_6, l_9)$ and $Q^2 = (l_7)$. Then, the following recency numbers are computed for all five locations encountered in the two routes. Going through the route Q^1, we compute $R(l_8|()) = \infty$, $R(l_9|(l_8)) = \infty$, $R(l_6|(l_8, l_9)) = \infty$, and $R(l_9|(l_8, l_9, l_6))) = 2$. For the second route Q^2, a single recency number is computed $R(l_7|()) = \infty$. Thus, the maximum finite recency number $r_{\max} = 2$.

For each recency number r between 1 and $r_{\max}$, we can compute the ratio of the number of times r actually happened to the number of times r could have possibly happened. Mathematically, this quantity is

$$P^\star(r) = \frac{\sum_{j=1}^{J} \left| \{i \mid R(q_i^j \mid (q_1^j, \ldots, q_{i-1}^j)) = r\} \right|}{\sum_{j=1}^{J} \max\{|Q^j| - r, 0\}}. \tag{7}$$

When r exceeds $r_{\max}$, the quantity $P^\star(r)$ is undefined. Continuing with the example above, we compute $P^\star(1)$ as the ratio of the number of times the recency number $r = 1$ was encountered in the training data, 0, to the number of times it could have possibly been encountered in the data, $\sum_{j=1}^{2} \max\{|Q^j| - 1, 0\} = 3 + 0 = 3$. Thus, $P^\star(1) = 0/3 = 0$. For the recency number $r = 2$, which was encountered once in the training data as $R(l_9|(l_8, l_9, l_6))) = 2$, we compute $P^\star(2) = 1/(2 + 0) = 0.5$.

After training, given a possible route length $n \in \mathbb{N}$, we define $x_n^\star = \min\{n, r_{\max}\}$. Then, the probability distribution over the recency numbers ($r \in \{1, 2, \ldots, x_n^\star, \infty\}$) is defined as

$$P_{\mathcal{R}}^n(r) = \begin{cases} P^\star(r), & 1 \le r \le x_n^\star \\ 1 - \sum_{k=1}^{x_n^\star} P^\star(k), & r = \infty. \end{cases} \tag{8}$$

Intuitively, given a route of n locations, $P_{\mathcal{R}}^n(r)$ gives the probability that the next, $(n+1)$th, location will yield the recency number r. Mathematically, for any given route length n, the function $P_{\mathcal{R}}^n(r)$ is indeed a probability distribution over $r \in \{1, 2, \ldots, x_n^\star, \infty\}$ insomuch as

$$\forall n \in \mathbb{N} \left[\sum_{r \in \{1, 2, \ldots, x_n^\star, \infty\}} P_{\mathcal{R}}^n(r) = 1 \right]. \tag{9}$$

Using the example data above, we will demonstrate computing the distribution $P_{\mathcal{R}}^n$ for $n \in \{1, 2, 3\}$. For $n = 1$, $x_n^\star = x_1^\star = \min\{1, 2\} = 1$. Thus, we can define the probability distribution $P_{\mathcal{R}}^1$ over $\{1, 2, \ldots, x_n^\star, \infty\}$ which becomes $\{1, \infty\}$. Specifically, $P_{\mathcal{R}}^1(1) = P^\star(1) = 0$ as we computed earlier. The second value of the distribution is $P_{\mathcal{R}}^1(\infty) = 1 - \sum_{k=1}^{x_1^\star} P^\star(k) = 1 - 0 = 1$. Intuitively, this means that for any route of $n = 1$ location, the probability of getting the recency number of 1 with the next (i.e., second) location is 0. This makes sense as the recency number of 1 means immediately repeating choice, and this never occurred in the training data. The probability mass is concentrated entirely over the recency number of ∞ is 1.

For $n = 2$, $x_n^\star = x_2^\star = \min\{2, 2\} = 2$, which allows us to define $P_{\mathcal{R}}^2(r)$ for r in $\{1, 2, \ldots, x_2^\star, \infty\} = \{1, 2, \infty\}$. As before, $P_{\mathcal{R}}^2(1) = P^\star(1) = 0$. For $r = 2$, we have $P_{\mathcal{R}}^2(2) = P^\star(2) = 0.5$. Finally, for $r = \infty$, we have $P_{\mathcal{R}}^2(\infty) = 1 - \sum_{k=1}^{x_2^\star} P^\star(k) = 1 - (0 + 0.5) = 0.5$.

The last example is for $n = 3$, which exceeds $r_{\max} = 2$. Thus, $x_n^\star = x_3^\star = \min\{3, 2\} = 2$, which allows us to define $P_{\mathcal{R}}^3(r)$ for r in $\{1, 2, \ldots, x_3^\star, \infty\} = \{1, 2, \infty\}$. The calculations are the same as in the previous paragraph, leaving us with the probability mass evenly divided between the recency numbers of 2 and ∞: $P_{\mathcal{R}}^3(2) = P_{\mathcal{R}}^3(\infty) = 0.5$. As before, $P_{\mathcal{R}}^3(1) = 0$.

d) Using the distributions: We defined three pdfs $P_{\mathcal{D}}, P_{\mathcal{A}}, P_{\mathcal{R}}^n$, which model the training data (i.e., location sequences recorded in the training environment). However, our AI agent is required to operate in a novel test environment for which it lacks any training data and, consequently, has no distributions. To solve this problem, we *port* the three distributions to the novel environment in the following fashion.

Suppose an agent hiding or seeking in a novel test environment has already visited n locations $Q = (q_1, \ldots, q_{n-1}, q_n)$. Then, strategy L2 stochastically selects the location l as the next location q_{n+1} with the probability $P_{\mathcal{D}}(i) \times P_{\mathcal{A}}(j) \times P_{\mathcal{R}}^n(R(l \mid Q))$ where the quantities i and j are defined below.

Let there be $|D|$ distinct interlocation distances in the training environment and $|D'|$ distinct interlocation distances in the test environment. Then, given the latest already selected location q_n and a candidate location l, we first compute the distance between them $E(l, q_n)$ and then take its rank $i' = I_{D'}(E(l, q_n))$ among all pairwise distances in the test environment. Then, we

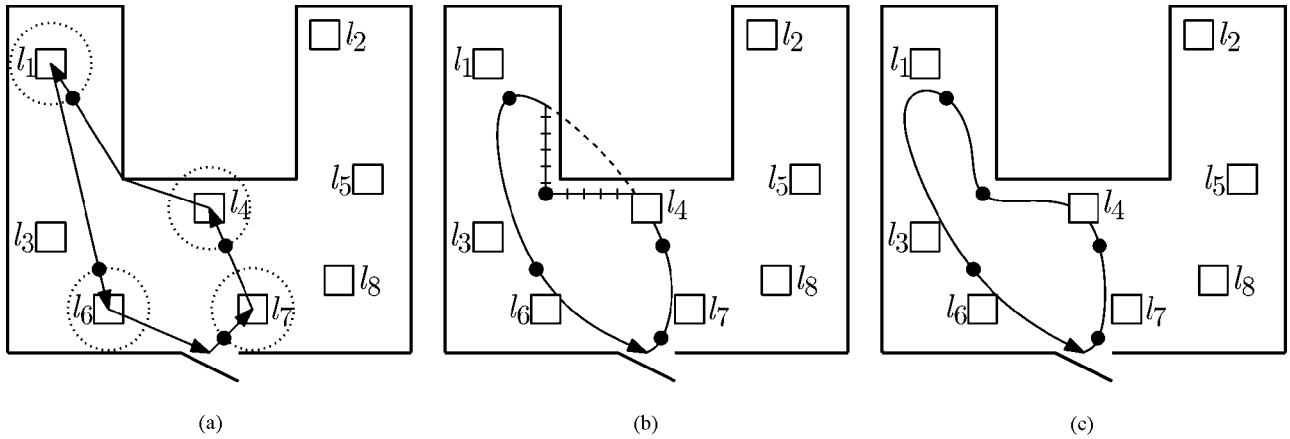


Fig. 5. Movement strategy M1.

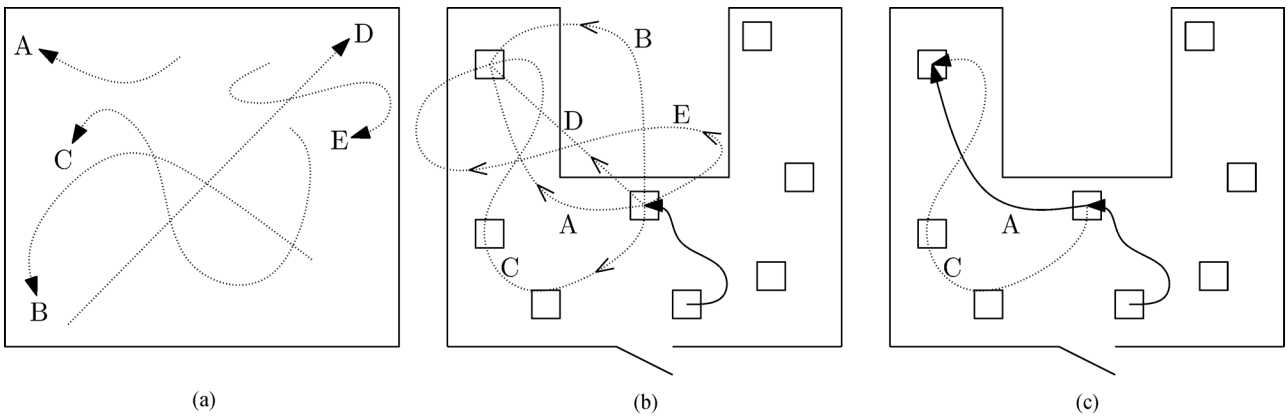


Fig. 6. Movement strategy M2.

linearly scale the rank i' to the range of the training environment: $i = 1 + (i' - 1)|D| - 1/|D'| - 1$. The result i is in the range $[1, |D|]$ and becomes an input to the distance distribution for the training environment: $P_{\mathcal{D}}(i)$.[3] Note that, for the first location to be selected, the expression $E(l, q_n)$ is not defined as there is no preceding location q_0. In that case, q_0 is artificially set to be the entry point to the environment (e.g., room's door).

A similar scaling and interpolation procedure is applied to the angle distribution $P_{\mathcal{A}}$. Specifically, the quantity j' is the rank of the angle $G(l, q_n, q_{n-1})$ among all $|A'|$ distinct angles possible, induced by the locations in the test environment. Mathematically, $j' = I_{A'}(G(l, q_n, q_{n-1}))$, which puts j' in the set $\{1, \ldots, |A'|\}$. We scale it linearly to place it in the range $[1, |A|]$ of angle ranks for the training environment: $j = 1 + (j' - 1)|A| - 1/|A'| - 1$. The resulting scaled rank j becomes the input to $P_{\mathcal{A}}$, with linear interpolation used to handle noninteger values of j. Note that the expression $G(l, q_n, q_{n-1})$ is undefined in two cases: when $n < 2$ and when the angle among l, q_n, q_{n-1} is ill-defined (i.e., when two of the three locations coincide). In either case, we set $P_{\mathcal{A}}(j) = 1/|L'|$ where L' is the set of locations in the test environment.

No scaling is required for the input to the recency distribution $P^n_{\mathcal{R}}$ since it does not directly depend on the environment geometry. However, we have a special case when the recency number (finite or infinite) for the candidate location l exceeds any finite recency number observed in the training data: $R(l|Q)wr_{\max}$. Then, we set the probability of selecting the location l as the $(n + 1)$th choice as follows. We compute the probability mass of the recency number of infinity scaled by the number (Ω) of all test locations whose recency number exceeds $r_{\max}$. Mathematically, $P^n_{\mathcal{R}}(R(l \mid Q)) = P^n_{\mathcal{R}}(\infty)/\Omega$ where $\Omega = |\{l' \in L' \mid R(l' \mid Q) > r_{\max}\}|$.

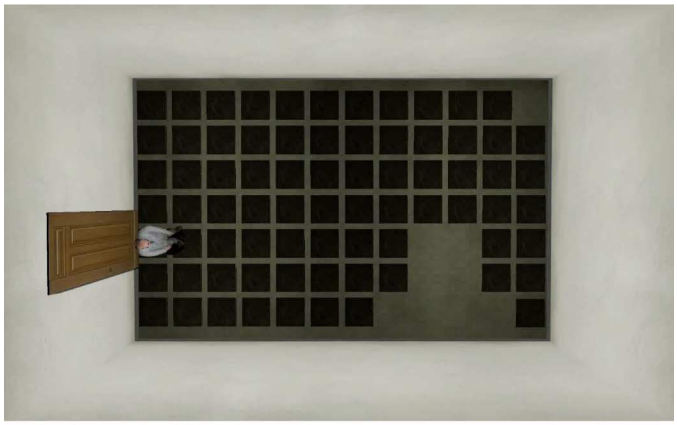
[3]Note that the distribution $P_{\mathcal{D}}$ for the training environment was defined only for the discrete inputs $\{1, \ldots, |D|\}$. We use linear interpolation to extend $P_{\mathcal{D}}$ to all points in the continuous interval $[1, |D|]$. This is done because the scaled rank i can be anywhere in $[1, |D|]$ and not just in the set $\{1, \ldots, |D|\}$.

B. Movement Strategies

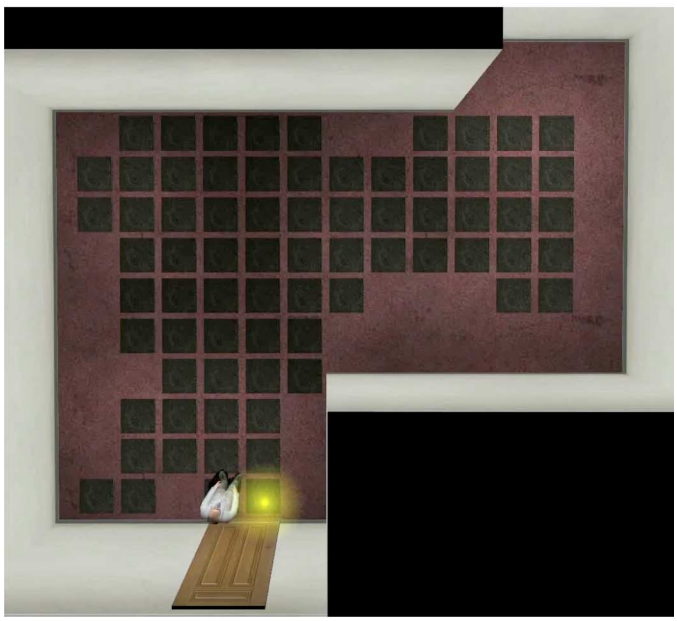
Once a sequence of locations is selected using the strategies L1 or L2, a movement strategy is used to navigate between them. We designed two movement strategies as follows.

1) Strategy M1 (Spline Interpolation): This strategy starts with the desired sequence of locations to be visited. In the example in Fig. 5, the agent begins at the door and intends to visit the locations l_7, l_4, l_1, and l_6 and then return to the door. First, strategy M1 runs A* [49] on a rectangular grid overlaid on the environment. As a result, it gets a path connecting the locations [Fig. 5(a)].

Next, M1 replaces each location in the sequence with a guide point. The guide point is the earliest point on the path from which the agent can access the location (i.e., perform hiding or seeking at that location). This was done because humans usually perform the hide-and-seek action at a selected location as soon



(a)



(b)

Fig. 7. Top–down view of the two environments. (a) Room A, a simple rectangular room. (b) Room B, an office-style room modeled after an existing laboratory. The rooms had 73 and 75 hide-and-seek locations, respectively.

as they get close enough to it. In Fig. 5(a), we show the "close enough" radii as the dashed circles around the locations and the guide points as the black dots at the intersection of the A* path and the circles.

M1 then fits cubic splines between the guide points to smooth the path. The resulting smoothed path is checked for intersecting obstacles and walls in the environment. If it does [as shown in Fig. 5(b)], then a new A* search is run between the guide points on each side of the intersection. The middle of the resulting path becomes a new guide point, and the spline fitting is conducted again. The process is repeated until all splines fit inside the environment and do not intersect any obstacles/walls. The final path is shown in Fig. 5(c). This process produces a smooth trajectory visiting the location sequence selected by strategy L1 or L2. The agent always faces the direction of its velocity vector.

2) Strategy M2 (Data-Driven Route Fitting): For this strategy, we first constructed a database of movements recorded in the training data. To do this, we took each route (e.g., Fig. 1) and split it into segments between each consecutive location

TABLE II
AGENTS A1 THROUGH A4

	Random selection (L1)	**Data-driven selection (L2)**
Spline-interpolation movement (M1)	Agent A1	Agent A2
Data-driven movement (M2)	Agent A3	Agent A4

selected by the participant [Fig. 6(a)]. Each segment was stored in the database with each point's coordinates (including the timestamps) recorded relative to the start of the segment.

This database was then used to compute the AI agent's paths in the test environment. Specifically, given a sequence of locations $(l_1, \ldots, l_n)$ to visit, the agent considered all consecutive location pairs $(l_i, l_{i+1}), i < n$. For each such pair, strategy M2 translated, rotated, and linearly scaled every segment in the database so that it connected l_i and l_{i+1}. An example of such fitted segments for a particular pair of locations is shown in Fig. 6(b). All segments not fully contained within the environment were discarded.[4] All remaining segments were then evaluated for their quality of fit—a product of the scaling fit and the continuity fit, described in detail in Section V-B. In Fig. 6(c), the segment A is fitter than the segment C because it is scaled less than C and has less angular difference from the previous segment.

Once all segments connecting the pair of locations l_i and $l_{i+1}, 1 \leq i < n$, were evaluated for their fit quality, a uniform random selection from the top ten fittest segments was used as the path between the locations. The stochasticity was introduced to add diversity to M2 and, consequently, make the agent less predictable while still using scaled, rotated, and translated training (i.e., human) paths.

C. Agent Structure

The four possible combinations of these strategies, (M1, L1), (M1, L2), (M2, L1), and (M2, L2), gave us the four agents (A1 through A4) shown in Table II. The agents were of different complexity and had different performance as discussed in Section V. Note that all four agents are designed to operate in a novel environment for which they have no training (i.e., human) data available.

We have also developed two special agents: A5 and A6. Agent A5 was meant to give away its artificial nature by purposely displaying nonhuman-like behavior. It was meant to study performance of the judges when such behavior is presented. Agent A6 was designed to behave like A5 at first but then switch to A4 behavior. It was meant to check if judges can be tricked by displaying obvious nonhuman behavior at first, followed by more humanlike behavior.

V. EMPIRICAL EVALUATION

In this section, we present an empirical evaluation of our model by running the agents A1–A6 through our specialized version of the Turing test. We implemented the hide-and-seek

[4]If no segments remained, then the agent fell back on strategy M1 for that pair of locations.

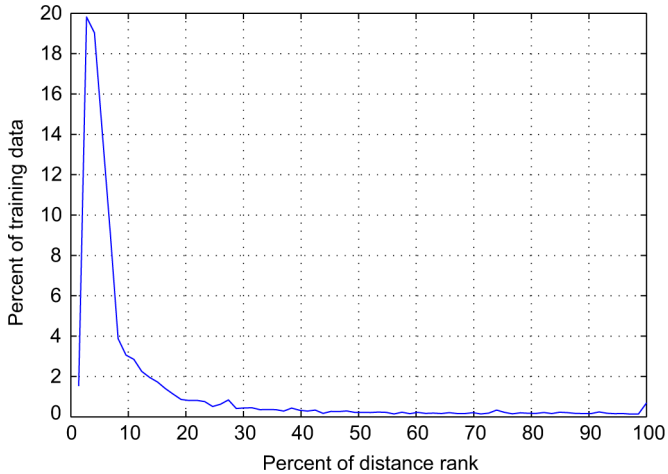


Fig. 8. $P_{\mathcal{D}}$ from the training data for the seek task in room B.

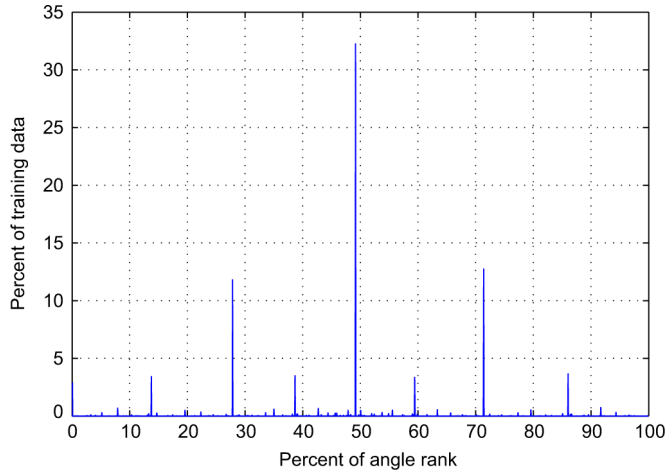


Fig. 9. $P_{\mathcal{A}}$ from the training data for the seek task in room A.

environments using the *Source* engine and the art assets of *Half-Life 2* [50]. Two environments shown in Fig. 7 were built using the *Hammer* editor [51]. The locations are represented by black floor tiles.

We then ran a group of human participants in the environments and recorded their routes while both hiding and seeking (Section V-A). The collected data were used to develop agents A1–A6 (Section V-B). Finally, another group of human participants were recruited as the judges (Section V-C).

A. Data Collection

The data set we used to create our models was the recordings of human participants in the two virtual environments. Our participants were recruited from first-year psychology courses and received a partial course credit for the participation.

The subjects were first trained to use the keyboard-and-mouse controls of *Half-life 2* by performing hiding and seeking in a separate training room. Then, they were asked to perform a hide task or a seek task in one of the two environments. In the hide task, the participants were asked to hide three objects and to "make your objects difficult for other people to find." In the seek task, the participants were asked to select locations until three previously hidden items were found. The seek task was limited to one minute, while the hide task was not time limited. Participants were free to move about the room, but had to wait for one second between selecting locations.

As the subjects performed their tasks, their avatar's locations and orientations were recorded once per second. Additionally, each location selection and the time at which the selection occurred were recorded. Overall, 5142 paths were recorded, each containing between 6 and 125 data points. They constituted the training data for the agent development.

B. Agent Development

As described in Section IV, strategy L1 selected locations uniformly randomly while strategy L2 used the training data collected by observing human behavior in both rooms. Under L2, three pdfs $P_{\mathcal{D}}$, $P_{\mathcal{A}}$, $P^n_{\mathcal{R}}$ were built. Several observations can be drawn from their shape as follows.

First, humans tend to prefer making selection choices near their current position (Fig. 8). This appears intuitive as accessing floor locations around the current position is easier and faster for it requires less travel. Second, the highest peak in angle rotation is 0° which, in Fig. 9, corresponds to the 50th percentile in the ranking spanning the range of $-180°$ to 180°. The two secondary peaks correspond to $-90°$ and 90° rotation angles (i.e., the two right angle turns). This is likely due to the regular placement of the locations on a rectangular grid (Fig. 7) and the preference of some humans to traverse the said grid systematically, row by row or column by column. Third, the more recently a location was selected the less likely humans were to select it again (Fig. 10) although there was a tendency to select the very same location twice in a row. This tendency is possibly due to the human participants' missing the selection confirmation message and, thus, reclicking on the floor tile right after selecting it.

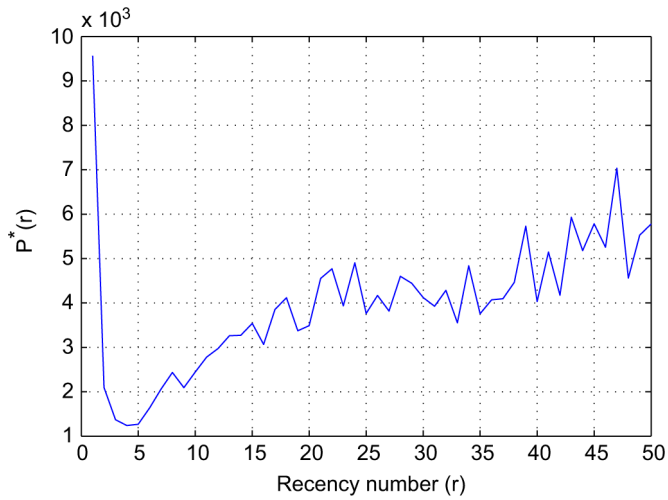


Fig. 10. Values of $P^*(r)$ for the first recency numbers ($r \in \{1, \ldots, 50\}$). The maximum finite recency number was $r_{\max} = 74$.

For the movement strategy M2, the scaling quality and the continuity quality were defined as follows. Suppose a segment of length a was fitted between two locations b distance apart (by shortest path). Then, the scaling quality of the fit is $\min\{a/b, b/a\}$. Intuitively, a segment that needs to be shrunk to half of its original length to fit between given locations is of the same fit quality as the segment that needs to be enlarged two times. The continuity quality of fitting a line segment was

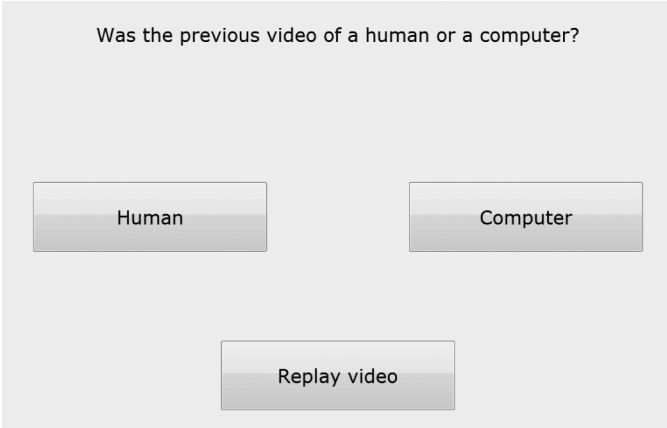


Fig. 11. Judge's labeling interface.

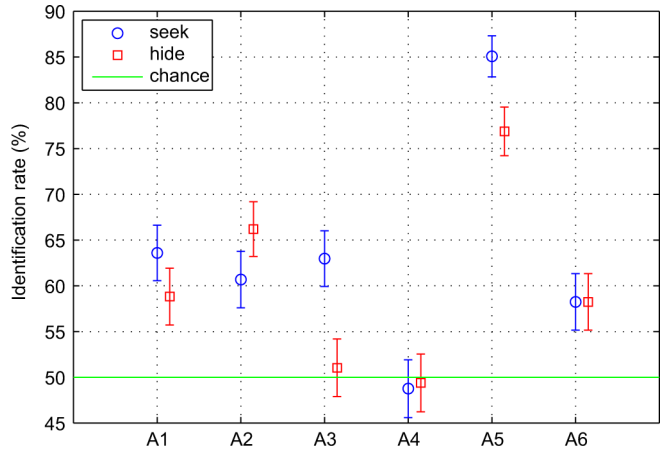


Fig. 12. Ninety five percent confidence intervals for identification rates divided by task (hide/seek).

computed as $1 - \Delta^\circ/180^\circ$ where Δ is the absolute value of the difference in the agent's heading angle at the end of the previous segment and the beginning of the segment being fit.

The two additional agents A5 and A6 were implemented as follows. Agent A5 picked two neighboring locations in a corner of the environment, stood between them so that it could access both of them without moving, and then alternately selected them (i.e., hid or sought at them). While doing so, it rotated in place with a constant angular speed. This was meant to give away its nonhuman nature. Agent A6 was equivalent to A5 for the first two routes and to agent A4 for the following routes. This was done to investigate whether the judges could be affected by demonstrating nonhuman-like behavior first, followed by humanlike behavior.

C. Judging

We recruited 288 human participants from first-year courses in psychology as our judges. This group did not overlap with the one described in Section V-A. The judges were divided into six groups of 48, one group for each AI agent design (A1–A6). Each of the six groups was further divided into two subgroups of 24, for the two possible room orderings (room A followed by room B or room B followed by room A).

Human and computer hide-and-seek behavior was presented to the judges in the form of top–down video recordings of routes [Fig. 7(b)]. In the figure, the agent is standing by the door having just accessed the location immediately on its right, indicated with a yellow glow. Each judge watched five training recordings followed by ten test recordings with no overlap between training and test recordings. The judges knew that the training recordings were of humans. The judges were told that some test recordings could be of an AI agent and some could be of humans but no specific proportion was given. In actuality, the proportion was approximately half. The judges were not told that there could be multiple agent designs. Each recording, including the training recordings, was played at double speed, lasting between 3 and 44 s. At the end of each test recording, the judges labeled it as "human" or "computer" using the interface shown in Fig. 11 which cued in the next recording. The judges were also able to rewatch the most recent recording multiple times before labeling it. In total, each judge labeled 40 recordings: 20 for the hide task and 20 for the seek task. Each judge's identification rate was calculated as the percentage of correct labelings. The judges were not informed of their identification rates.

VI. Study Results

The mean identification rates across all judges are shown in Figs. 12–14, together with 95% Wilson confidence intervals. If such a confidence interval lies completely below another interval, we can conclude that the previous agent is correctly labeled less often than the latter agent with greater than 95% confidence. The identification rates are divided by task (hide/seek), by agent (human/AI), and by judge experience (low/high). We will analyze them below.

A. Hiding Versus Seeking

Fig. 12 shows agent identification rates for the hide-and-seek tasks. For the hide task, AI agents using the data-driven movement strategy appear significantly more humanlike than agents using the spline-based M1 strategy. Indeed, the 95% confidence interval for A3 (M2, L1) is below the interval for A1 (M1, L1), and the interval for A4 (M2, L2) is below the interval for A2 (M1, L2). On the other hand, data-driven selection strategy L2 is not significantly different from the random location–selection strategy L1.

For the seek task, the AI agent A4 was judged significantly more humanlike than the agents A1, A2, and A3 (which are not significantly different from each other). L2 strategy produced no significant difference in the hide task, but did so in the seek task when paired with M2 strategy (agent A4 was judged significantly more humanlike than agent A3). The location–selection strategy appears to play a smaller part in the hide task because its low number of location choices (three) may not suffice to judge a route as human or AI.

B. Humans Versus AI Agents

Fig. 13 shows judge identification rates for human versus AI agent routes. Note a general trend that the identification rate of human routes rose when such routes were presented in the same batch with weaker agents and fell when mixed with stronger

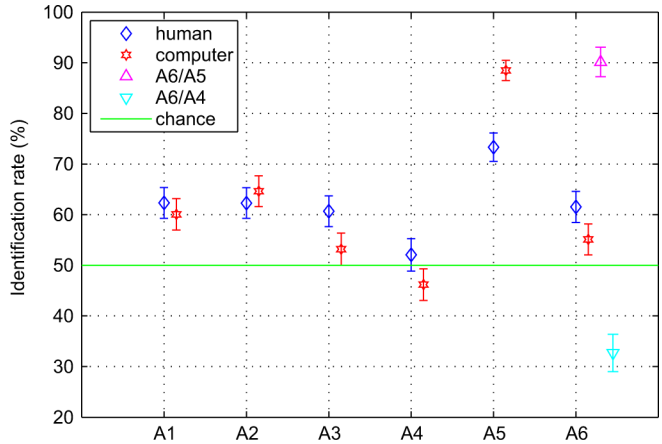


Fig. 13. Ninety five percent confidence intervals for identification rates presented by the agent type. Two additional intervals are shown: "A6/A5" shows the first two routes of A6 (when it is equivalent to A5). "A6/A4" shows the last three routes of A6 (when it is equivalent to A4).

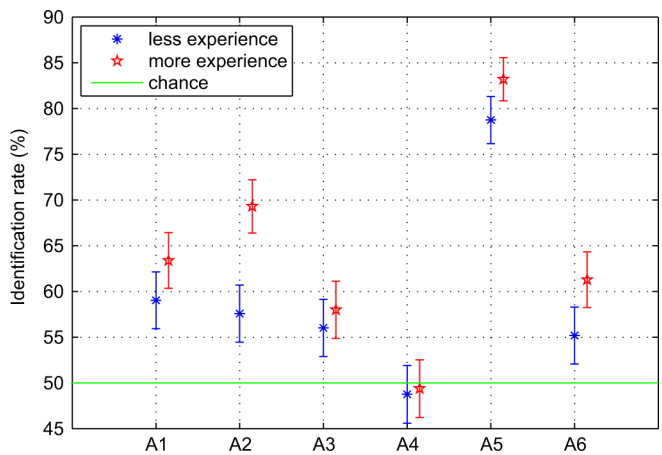


Fig. 14. Ninety five percent confidence intervals for identification rates by judge experience.

agents. The 95%-confidence interval for the human identification rate when paired with agent A4 was below the human intervals of A1, A2, and A3. Similarly, the human interval for A5 was above such for A1, A2, and A3. Agents A4 and A5 were designed to be the strongest and weakest agents, respectively (which is supported by the positions of their computer intervals in the figure).

The fact that judges' success in identifying humans as humans appears to depend on the type of the AI agent presented in the same batch is curious. To explain it, we conjecture that human judges tend to assume an equal mixture of routes in a batch. Specifically, if there are ten routes a judge is to label as "human" or "computer," then he/she will tend to assign five "human" and five "computer" labels. This is despite the fact that the judges were never told that there is an expected balance in the routes presented to them.

Agent A6 generated an intentionally nonhuman route (using A5) as the first two routes it was run for, and then used the most humanlike design (A4) for the following three routes it was run for. Most judges correctly labeled the first two routes as "computer" (the mean identification rate for A5 is 88.5%). The remaining eight routes (three produced by agent A4 and five produced by actual humans) appear all humanlike. By guessing randomly but maintaining the equal balance proportions (i.e., labeling random three out of the eight routes as "computer" and the remaining five routes as "human") the judges would have an expected identification rate of 37.5% for the computer (A6, last three routes) and 62.5% for humans. The empirical means of 35.3% and 61% appear to support this conjecture.

C. *Effects of Practice*

In Fig. 14, we investigate the effects of practice on judges' performance. We show the identification rates broken down by judge experience. Each judge labeled 20 routes in each of the two rooms (the room order was counterbalanced). In the figure, the "less experience" intervals show 95% confidence intervals for judge performance for different agent types averaged over the first 20 routes across all judges. The "more experience" confidence intervals are for the last 20 routes.

While the judges did not receive any feedback on their labeling, their performance appears to improve with practice for all AI agent types, but the amount of improvement varies.

VII. Conclusion and Future Work

Hiding and seeking are important cognitive abilities of humans and animals and have several applications in video game design. This paper made the following contributions. We proposed the first computational generative model designed to mimic people's hide-and-seek behavior in a virtual room. The model is built automatically from observed human hide-and-seek behavior. We implemented a model within an AI agent and demonstrated its validity via a restricted version of the Turing test. Specifically, four AI agents based on the model were constructed and evaluated. The most complex of the four agents used machine-learned patterns of human behavior and appears to have passed the restricted version of the Turing test. Specifically, the 95% confidence intervals for A4 fall within the range of 45% –55% for both hide-and-seek tasks. In other words, we are 95% confident that the identification rate of human judges for agent A4 is within 5% of chance.

Future work will investigate applicability of this model to other animals and other spatial tasks. Furthermore, while this study used only sparsely populated geometrically simple rooms in the empirical evaluation, future work will investigate the extent to which the agent designs evaluated in this paper manage to capture human behavior in environments with more varied hiding locations and the higher visual fidelity found in many commercial video games. If this is successful, then our agents can be incorporated into a video game to take advantage of predicting player's search patterns.

Acknowledgment

The authors would like to thank the Intelligent Reasoning Critiquing and Learning (IRCL) research group: D. Thue, D. Huntley, and G. Lee, for fruitful discussions. They would also like to thank Valve Software Corp. (Bellevue, WA, USA) for developing software and releasing their toolset. Finally, the authors appreciate feedback from the reviewers.

REFERENCES

[1] K. J. Talbot, E. L. Legge, V. Bulitko, and M. L. Spetch, "Hiding and searching strategies of adult humans in a virtual and a real-space room," *Learn. Motivat.*, vol. 40, pp. 221–233, 2009.

[2] S. Moffat and S. Resnick, "Effects of age on virtual environment place navigation and allocentric cognitive mapping," *Behav. Neurosci.*, vol. 116, pp. 851–859, 2002.

[3] Valve Software, Counter-Strike: Source. 2008 [Online]. Available: http://store.steampowered.com/app/240/

[4] Gearbox Software, Borderlands. 2009 [Online]. Available: http://www.borderlandsthegame.com/

[5] Bethesda Softworks, Fallout: New Vegas. 2011 [Online]. Available: http://fallout.bethsoft.com/

[6] Blizzard Entertainment, Starcraft 2. 2010 [Online]. Available: http://www.starcraft2.com/

[7] A. Cenkner, V. Bulitko, and M. L. Spetch, "A generative computational model for human hide and seek behavior," in *Proc. 7th Artif. Intell. Interactive Digit. Entertain. Conf.*, Palo Alto, CA, USA, 2011, pp. 128–133.

[8] A. Turing, "Can automatic calculating machines be said to think?" The Turing Digital Archive, 1952 [Online]. Available: http://www.turingarchive.org/browse.php/B/6.

[9] J. Togelius, G. N. Yannakakis, S. Karakovskiy, and N. Shaker, "Assessing believability," in *Believable Bots*, P. F. Hingston, Ed. Berlin, Germany: Springer-Verlag, 2012, pp. 215–230.

[10] H. Loebner, "Loebner Prize in artificial intelligence," Nov. 2011 [Online]. Available: http://www.loebner.net/Prizef/loebner-prize.html

[11] P. Hingston, "A Turing test for computer game bots," *IEEE Trans. Comput. Intell. AI Games*, vol. 1, no. 3, pp. 169–186, Sep. 2009.

[12] 2K Games, The 2K BotPrize Nov. 2011 [Online]. Available: http://botprize.org/

[13] N. Shaker, J. Togelius, and G. Yannakakis, "The Turing test track of the Mario AI Championship," Aug. 2012 [Online]. Available: http://www.marioai.org/turing-test-track

[14] N. Emery and N. Clayton, "The mentality of crows: Convergent evolution of intelligence in corvids and apes," *Science*, vol. 306, pp. 1903–1907, 2004.

[15] F. D. Rose, E. A. Attree, B. M. Brooks, D. M. Parslow, and P. R. Penn, "Training in virtual environments: transfer to real world tasks and equivalence to real task training," *Ergonomics*, vol. 43, no. 4, pp. 494–511, 2000.

[16] N. Foreman, J. Stirk, J. Pohl, L. Mandelkow, M. Lehnung, A. Herzog, and B. Leplow, "Spatial information transfer from virtual to real versions of the Kiel locomotor maze," *Behav. Brain Res.*, vol. 112, no. 1, pp. 53–61, 2000.

[17] J. McComas, J. Pivik, and M. Laflamme, "Children's transfer of spatial learning from virtual reality to real environments," *CyberPsychol. Behav.*, vol. 1, no. 2, pp. 121–128, 1998.

[18] P. N. Wilson, N. Foreman, and M. Tlauka, "Transfer of spatial information from a virtual to a real environment in physically disabled children," *Disability Rehabil.*, vol. 18, no. 12, pp. 633–637, 1996.

[19] B. G. Witmer and P. B. Kline, "Judging perceived and traversed distance in virtual environments," *Presence*, vol. 7, no. 2, pp. 144–167, 1998.

[20] E. Kalová, K. Vlcek, E. Jarolmová, and J. Bures, "Allothetic orientation and sequential ordering of places is impaired in early stages of Alzheimer's disease: corresponding results in real space tests and computer tests," *Behav. Brain Res.*, vol. 159, no. 2, pp. 175–186, 2005.

[21] A. Schmelter, P. Jansen, and M. Heil, "Empirical evaluation of virtual environment technology as an experimental tool in developmental spatial cognition research," *Eur. J. Cogn. Psychol.*, vol. 21, no. 5, pp. 724–739, 2009.

[22] T. Hartley, E. A. Maguire, H. J. Spiers, and N. Burgess, "The well-worn route and the path less traveled: distinct neural bases of route following and wayfinding in humans," *Neuron*, vol. 37, no. 5, pp. 877–888, 2003.

[23] V. D. Bohbot, J. Lerch, B. Thorndycraft, G. Iaria, and A. P. Zijdenbos, "Gray matter differences correlate with spontaneous strategies in a human virtual navigation task," *J. Neurosci.*, vol. 27, no. 38, pp. 10 078–10 083, 2007.

[24] D. M. Kelly and W. F. Bischof, "Orienting in virtual environments: How are surface features and environmental geometry weighted in an orientation task?," *Cognition*, vol. 109, no. 1, pp. 89–104, 2008.

[25] S. A. Livingstone and R. W. Skelton, "Virtual environment navigation tasks and the assessment of cognitive deficits in individuals with brain injury," *Behav. Brain Res.*, vol. 185, no. 1, pp. 21–31, 2007.

[26] A. A. Rizzo, J. G. Buckwalter, J. S. McGee, T. Bowerly, C. v. d. Zaag, U. Neumann, M. Thiebaux, L. Kim, J. Pair, and C. Chua, "Virtual environments for assessing and rehabilitating cognitive/functional performance," *Presence*, vol. 10, no. 4, pp. 359–374, 2001.

[27] F. D. Rose, B. M. Brooks, and A. A. Rizzo, "Virtual reality in brain damage rehabilitation: Review," *CyberPsychol. Behav.*, vol. 8, no. 3, pp. 241–262, 2005.

[28] M. Wu, A. Liu, and K. M. Chandy, "Virtual environments for developing strategies for interdicting terrorists carrying dirty bombs," in *Proc. Int. Soc. Crisis Response Manage. Conf.*, 2008, pp. 1–5.

[29] E. L. Legge, M. L. Spetch, A. Cenkner, V. Bulitko, C. Anderson, M. Brown, and D. Heth, "Not all locations are created equal: Exploring how adults hide and search for objects," *PLoS ONE*, vol. 7, no. 5, 2012, DOI: 10.1371/journal.pone.0036993.

[30] N. Clayton, J. Dally, and N. Emery, "Social cognition by food-caching corvids. The western scrub-jay as a natural psychologist," *Philosoph. Trans. Roy. Soc. B, Biol. Sci*, vol. 262, pp. 507–522, 2007.

[31] N. Clayton, N. Emery, and A. Dickinson, "The prospective cognition of food caching and recovery by western scrub-jays (aphelocoma californica)," *Comparative Cogn. Behav. Rev.*, vol. 1, pp. 1–11, 2006.

[32] J. Dally, N. Clayton, and N. Emery, "The behavior and evolution of cache protection and pilferage," *Animal Behav.*, vol. 72, pp. 13–23, 2006.

[33] E. H. Cornell and C. D. Heth, "The spatial organization of hiding and recovery of objects by children," *Child Develop.*, vol. 57, pp. 603–615, 1986.

[34] E. H. Cornell, C. D. Heth, L. S. Broda, and V. Butterfield, "Spatial matching in 1 1/2- to 4 1/2-year-old children," *Develop. Psychol.*, vol. 23, pp. 499–508, 1987.

[35] A. Turing, "Computing machinery and intelligence," *Mind*, vol. 49, pp. 433–460, 1950.

[36] D. Livingstone, "Turing's test and believable AI in games," *Theor. Practical Comput. Appl. Entertain.*, vol. 4, pp. 6–18, 2006.

[37] D. Hirono and R. Thawonmas, "Implementation of a human-like bot in a first person shooter: Second place bot at botprize 2008," in *Proc. Asia Simul. Conf.*, Shiga, Japan, 2009.

[38] D. Wang, B. Subagdja, A. Tan, and G. Ng, "Creating human-like autonomous players in real-time first person shooter computer games," in *Proc. 21st Conf. Innovative Appl. Artif. Intell.*, Palo Alto, CA, USA, 2011, pp. 173–178.

[39] B. Tastan and G. Sukthankar, "Learning policies for first person shooter games using inverse reinforcement learning," in *Proc. 7th Artif. Intell. Interactive Digit. Entertain. Conf.*, Palo Alto, CA, USA, 2011, pp. 85–90.

[40] The 2k botprize: Result 2012 [Online]. Available: http://botprize.org/result.html

[41] M. Booth, "The official counter-strike bot," 2004 [Online]. Available: http://www.gdcvault.com/play/1013625/

[42] L. Lidén, "Strategic and tactical reasoning with waypoints," in *AI Game Programming Wisdom*. Newton Center, MA, USA: Charles River Media, 2002, pp. 211–220.

[43] C. Bererton, "State estimation for game AI using particle filters," AAAI Workshop, Tech. Rep. WS-04-04, 2004.

[44] S. Hladky and V. Bulitko, "An evaluation of models for predicting opponent locations in first-person shooter video games," in *Proc. IEEE Symp. Comput. Intell. Games*, 2008, pp. 39–46.

[45] C. Darken and B. Anderegg, "Particle filters and simulacra for more realistic opponent tracking," in *Game AI Programming Wisdom 4*. Newton Center, MA, USA: Charles River Media, 2008, pp. 419–428.

[46] F. Southey, W. Loh, and D. Wilkinson, "Inferring complex agent motions from partial trajectory observations," in *Proc. 20th Int. Joint Conf. Artif. Intell.*, San Francisco, CA, USA, 2007, pp. 2631–2637.

[47] S. Butler and Y. Demiris, "Partial observability during predictions of the opponent's movements in an RTS game," in *Proc. IEEE Comput. Intell. Games Symp.*, Copenhagen, Denmark, 2007, pp. 46–53.

[48] B. G. Weber, M. Mateas, and A. Jhala, "A particle model for state estimation in real-time strategy games," in *Proc. 7th Artif. Intell. Interactive Digit. Entertain. Conf.*, Palo Alto, CA, USA, 2011, pp. 103–108.

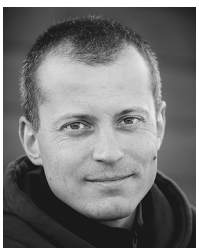
[49] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.

[50] Valve Corporation, Half-Life 2. 2004 [Online]. Available: http://www.half-life2.com/

[51] Valve Corporation, Hammer. 1996 [Online]. Available: http://developer.valvesoftware.com/wiki/Valve_Hammer_Editor/

Andrew Cenkner received the M.Sc. degree in computing science with a focus on artificial intelligence from the University of Alberta, Edmonton, AB, Canada.

His research interests are computing science and number theory. In particular, he is interested in the limits of computation, mimicking human behavior, and the prime number theorem.



Vadim Bulitko received the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign, Urbana, IL, USA, in 1999.

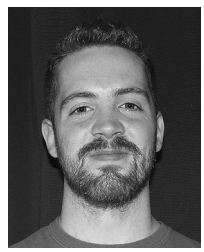
He is an Associate Professor at the Department of Computing Science, University of Alberta, Edmonton, AB, Canada. He is interested in building the strong artificial intelligence as well as understanding intelligence and cognition in humans and animals. He specializes in real-time heuristic search, AI in computer games, including interactive narrative, and cognitive processes and models.

Marcia Spetch received the Ph.D. degree in psychology from the University of British Columbia, Vancouver, BC, Canada, in 1982.

She is a Professor at the Department of Psychology, University of Alberta, Edmonton, AB, Canada. She investigates learning and cognitive processes in humans and other animals, with a current focus on spatial cognition and choice behavior.

Eric Legge received the Ph.D. degree in psychology from the University of Alberta, Edmonton, AB, Canada, in 2013.

He is interested in how human and nonhuman animals orient, navigate, and locate goals in natural and artificial environments. He specializes in cross-species comparisons of spatial navigation, the use of Bayesian inference in goal localization, and understanding how spatial strategies improve memory in humans.



Craig G. Anderson received the B.Sc. degree in psychology from the University of Alberta, Edmonton, AB, Canada.

He is interested in the cognitive effects of video games. He is particularly interested in the role of flow in learning, as well as the influence that video games have on society.

Matthew Brown received the Ph.D. degree in neuroscience from the University of Western Ontario, London, ON, Canada, in 2007.

He is a Research Associate at the Department of Psychiatry, University of Alberta, Edmonton, AB, Canada. He has expertise in machine learning and its application to cognitive neuroscience and brain imaging. He is interested in understanding risk-related decision making in humans and simulated agents.