A Generative Computational Model for Human Hide and Seek Behavior

Andrew Cenkner¹ and Vadim Bulitko¹ and Marcia Spetch²
Departments of Computer Science¹ and Psychology²
University of Alberta
Edmonton, AB, T6G 2E8, CANADA
cenkner | bulitko | mspetch @ ualberta.ca

Abstract
Hiding and seeking is a cognitive ability frequently demonstrated by humans in both real life and video games. We use machine learning to automatically construct the first computational model of hide/seek behavior in adult humans in a video game-like setting. The model is then run generatively in a novel environment and its behavior is found indistinguishable from actual human behavior by a panel of human judges. In doing so the AI agent using the model appears to have passed a version of the Turing test for hiding and seeking.

Introduction
Hiding and seeking is a cognitive ability of humans studied extensively in psychology (Talbot et al. 2009). Most video games invoke this ability in some form. For instance, competitive online first-person shooters such as Counter-strike: Source (Valve Software 2008) have players searching for members of the opposing team (e.g., snipers). Role-playing games such as Fallout: New Vegas (Bethesda Softworks 2011) or Borderlands (Gearbox Software 2009) encourage the player to explore the environment and reward them 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 loot (either statically or procedurally) 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 development stage, where the players will search and how their search patterns will be different depending on the player type (e.g., from a casual player to a completionist).

Second, level designers can also benefit from knowing a priori where the players will be looking for other players (e.g., in Counter-strike: Source) or other player’s units (e.g., in StarCraft 2 (Blizzard Entertainment 2010)). Finally, AI developers need to develop NPCs that search for the player in a compelling, non-cheating way. In Counter-strike: Source this was accomplished by hard-coding search patterns from developer’s experience (Booth 2004). Other games attempt to generate realistically looking seeking behavior by obfuscating AI’s omniscient knowledge of the player’s location. Both scenarios are expensive, error-prone and require extensive tuning and testing.

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 stashes of weapons, improvised explosive devices, etc.) Finally, if hiding and seeking are viewed as fundamental cognitive abilities then mastering them via a computer program/model may bring us closer to building strong Artificial Intelligence.

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

Problem Formulation
In this paper we consider the task of hiding and seeking in a three-dimensional virtual environment. A human or an AI agent selects from a finite pre-determined set of locations. They do so by navigating in the environment, pointing at a location and indicating whether they wish to hide or seek there (Figure 1).

This activity involves two parts: selecting locations and navigating there. A compelling AI agent has to master both to appear human-like. We say that a generative computational model of hide/seek behavior is successful if it can be used within an AI agent to pass the following version of Turing test in a novel room. What we mean by novel room is that the AI agent must have never seen the room or any data from human activities in the room, and act completely on knowledge it has gathered from different rooms in the past. If the AI agent was allowed to test on the same room it trained on it would not be generative.

Our version of Turing test involves a panel of human judges separately observing an equal mixture of human and AI agents moving about in an environment and picking locations to hide or seek. After each judge views an agent they decide if the agent is in fact a human or an AI program. Their binary votes are tallied and a detection rate is averaged over a number of demonstrated hide/seek behaviors and a number of judges. An AI agent is said to have passed the...
Turing test if the detection rate is statistically indistinguishable from chance. We then say that the model that the agent used is validated.

Related Work

Psychologists have long researched animal hiding and seeking behavior, in particular with respect to food caches (Clayton, Dally, and Emery 2007; Clayton, Emery, and Dickinson 2006; Dally, Clayton, and Emery 2006). There have also been a number of studies on hiding and seeking behavior in children (Cornell and Heth 1986; Cornell et al. 1987). None of these studies is directly applicable to our problem as they do not consider human adults.

A more recent study (Talbot et al. 2009) did consider hiding and seeking behavior of adult humans in a game-like virtual environment. Unfortunately, it does not solve our problem for two reasons. First, the environment used in the study — a small square room devoid of any features, with a total of nine hiding/seeking locations — was far simpler than a typical video game environment. Second, the behavior exhibited by humans was analyzed at a coarse aggregate level (e.g., mean distance traveled to the first location from the room entrance). No attempt to computationally generate hiding/seeking behavior was made.

Some work has been done in the area of predicting player locations in first-person shooters (Hladky 2009; Darken and Anderegg 2008). The work was done to predict human behavior in the task of hiding oneself and seeking for other players. In contrast our work deals with the task of participants hiding and seeking for arbitrary objects. They are not told what the objects are and are simply asked to “Make your objects difficult for other people to find.” Another difference is that after training a model on human behavior we create an agent that uses this model to act like the humans.

Proposed Approach

We set out to develop a computational model capable of effectively generating human-like hide and seek behaviors. The desired properties were as follows. First, we wanted our model to be developed automatically via data-mining previously collected recordings of human hide/seek behavior (we call this stage “training”). Second, we wanted the model to be portable — that is, applicable in an environment it has never seen before — without any extensive annotation of such a novel environment. Third, the model had to be capable of both selecting hide/seek locations and producing a realistic path between them. Fourth, the model was to be stochastic in nature and capable of producing many human-like behaviors in the same environment. Finally, the model was to produce both hiding and seeking behaviors.

We decided to model human behavior by breaking it into two smaller behaviors: how humans select locations and how humans move. We created both a simple and an advanced strategy for each of these behaviors. This was done for two reasons. First, we want to compare the advanced strategies to the simple strategies to demonstrate that our version of the Turing test is meaningful and cannot be passed trivially. Second, we want to gain insight into how important each behavior is when trying to act human. To elaborate on this with an example, we might find that using the advanced movement strategy helps more than the advanced location selection strategy.

Location Selection Strategies

For the simple location selection strategy (L1) we used uniform random selection and for the advanced location selection strategy (L2) we used a stochastic selection from a previously trained distribution. An agent would use its strategy (L1 or L2) to select three location for hiding or, when seeking, continue making selections until finding all objects or running out of time (one minute).

Strategy L1: random selection. Locations are selected uniformly randomly from a set of all possible locations. In the hiding task, three locations are picked without replacement. In the seeking task, locations are picked indefinitely with replacement from the set. We used replacement in the seeking task as humans were found to re-visit locations frequently which was not the case in the hiding task.

Strategy L2: data-driven selection. Locations are drawn randomly using the product of the following three distributions. The first distribution is of the straight line distances between consecutive location choices recorded in the training data. The second distribution is of the angles formed by every triple of consecutive locations in the training data. The final distribution modeled the frequency in which participants clicked on a tile they already clicked on.

Movement Strategies

Once a set of locations was picked using L1 or L2, a movement strategy was used to navigate between them. For the simple movement strategy (M1) we used smooth spline interpolation between the desired locations and for the advanced movement strategy (M2) we searched a library of
human movements and selected one that fit the desired path well.

**Strategy M1: spline interpolation.** Given a set of locations, a cubic spline interpolation was used to construct a smooth path through them. If a segment between points $a$ and $b$ of the resulting path intersected an obstacle in the environment then $A^*$ was used to construct a new valid path between $a$ and $b$ and its middle point was inserted in the set of locations between $a$ and $b$. The spline fitting was then repeated for the new set of locations. The process stopped when the resulting path fit the environment. This strategy is complete in the sense that given enough iterations it will just degenerate into $A^*$. In other words if there is a path between $a$ and $b$ this strategy will find one.

**Strategy M2: data-driven trajectory shaping.** Given a set of locations $\{l_1, \ldots, l_n\}$ to traverse, we considered them sequentially. For each pair of locations $(l_i, l_{i+1})$, we translated, rotated and stretched every recorded trajectory in the training data so that it connected $l_i$ and $l_{i+1}$. The quality of each such fit was determined by a linear combination of the amount of stretching required and the difference in the agent’s actual orientation at location $l_i$ and the orientation resulting from fitting the path. A stochastic selection from the best fitting paths was used for navigation.

**Agents**

To test the effectiveness of these models we created an agent for each of the four possible combinations of choosing simple or advanced in both behaviors. The breakdown of the models used for each agent can be seen in Table 1.

<table>
<thead>
<tr>
<th>Spline:M1</th>
<th>Human Trained:M2</th>
<th>Agent1:A1</th>
<th>Agent2:A2</th>
<th>Agent3:A3</th>
<th>Agent4:A4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random:L1</td>
<td>Human Trained:L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Definition of agents with movement strategies on left and location selection strategies on top.

As the subjects performed their tasks, the $x, y, z$ coordinates as well as $\theta, \phi$ angles of their avatar were recorded with the frequency of 1 Hz. Additionally, their location selections were recorded as well, whenever they clicked on a tile. Overall, with each participant running multiple hide and seek tasks, more than 5142 vectors were recorded, each containing between 6 and 125 data points.

**Empirical Evaluation**

In our evaluation we pursued two objectives. First we aimed to demonstrate that the version of Turing test we used is indeed meaningful. This involves showing that (i) the task is cognitively rich enough that a simple AI agent would fail the test and (ii) that the human judges are given enough information to render an informed judgement. An example of a test that violates condition (i) would be “to sit in the chair” since a statically placed agent model would easily pass the test. An example of violating condition (ii) would be withholding the actual contents of a chat from the judge in the original Turing test and, instead, showing them only a light when the agent is using their teletype.

We satisfy condition (ii) by presenting the judges with a video of a top-down view of the agent (human or AI) moving about in the environment and selecting locations. Preliminary tests clearly demonstrated that this input is rich enough that judges can easily tell between a human agent and an AI agent that selects locations at random and navigates between them along a shortest path. We satisfy condition (i) by showing that a non-trivial, manually designed AI agent can be reliably distinguished from humans from such videos thereby failing the test.

**Experimental Procedure**

The study was carried out in three parts: data collection, model/agent training and judging.

**Data Collection.** The dataset we used to create our models was the collective recordings of over 1071 human participants in virtual environments. The participants were recruited from a first-year course in psychology. We were able to obtain such a large number of participants because the dataset is also being used in other experiments. Each participant was briefed with a description of the task and trained on how to control a first-person avatar with the Half-Life 2 mouse-and-keyboard controls (Valve Corporation 2004) in a small specifically designed training environment. The participants were then put in one of the two virtual environments built with Hammer (Valve Coorporation 1996) and executed with Source engine (Valve Corporation 2007) and asked to pick three hiding locations (the “hide task”) or pick locations until three previously hidden items are found (the “seek task”). In the latter case there was no limit on the number of locations but the time was limited to 1 minute.

The environments were: an office style room modeled after an existing laboratory and a simple rectangular room (Figure 2). The former contained 73 designated locations for hiding and seeking, shown as black floor tiles. The latter had 75 locations. Both environments had realistic lighting and office furniture.

As the subjects performed their tasks, the $x, y, z$ coordinates as well as $\theta, \phi$ angles of their avatar were recorded with the frequency of 1 Hz. Additionally, their location selections were recorded as well, whenever they clicked on a tile. Overall, with each participant running multiple hide and seek tasks, more than 5142 vectors were recorded, each containing between 6 and 125 data points.

**Model/Agent Training.** Each agent first creates a list of location selections using either L1 or L2. In the hide task a list of 3 selections is made and in the seek task a list of selections is created that is long enough to ensure the tasks timeout, 1 minute, will happen before reaching the end. The list of selections is then passed to the movement model. The
movement model, M1 or M2, then fits a path from the player starting area, near the room’s door, within clicking distance of each sequential location in the list and back to the door. The L2 and M2 models use the collected human data, while the L1 and M1 models do not.

The L1 model is very simple and just makes it’s next selection for the list by choosing a uniformly random location. The hide task selects without replacement and the seek task selects with replacement. In other words in the L1 model every location is equally likely to be picked as the next selection in the list, except if that location has already been selected and the agent is performing the hide task.

The first attempt at creating the M1 model was to simply fit a cubic spline through all the 3-space coordinates of the given list of locations. This proved to be a bad approach as the agent ended up standing right on top of each location before clicking on it. In practice most participants do not even walk over their selected location after selecting it, and almost none stood directly on top of the location before selecting it. To remedy this problem instead of making M1 fit a spline through the actual selection locations we made it fit a spline through points beside the actual locations. The points beside the the actual locations were created by running an A* path through the actual locations and taking the first point along the A* path within a reachable distance of the actual location. In other words M1 doesn’t quite travel to each location selection and instead comes within a reachable distance and starts heading to the next location. The cubic spline interpolation was done by parameterizing all of the desired player locations with respect to time and performing a one dimensional cubic spline interpolation on each one.

The L2 model trains three separate probability distributions from the human data provided. The first distribution is how likely each location is to be clicked given its distance from the last location. The goal of the model is to be generative, so we had to come up with a distribution that could scale to new environments. For example if we simply created a distribution for a small room with a maximum distance of \( k \) and then we tried to apply it to an open outdoor environment with a minimum distance greater than \( k \) the distribution would be useless as it would only say there should be a 100% chance of hiding at a distance less than \( k \).

To fix this we ranked the tiles in order of straight line distance. If there are \( n \) tiles in the environment there are \( n \) bins created in the distribution. Every time someone hides in the \( i \)th furthest tile it contributes to the \( i \)th bin. In this way the probability that the furthest tile is selected maps to the furthest tile being selected in the new environment and so on. The second distribution created was for the angle needed to rotate to select the location. This distribution was also created on a ranking (from the left most location begin the lowest to the right most being the highest) so that even strangely placed locations will still scale to the distribution. The third distribution was the probability a tile is clicked given that it was clicked \( k \) times ago. All the locations start with a value of infinity (they were never clicked before). This distribution is clearly generative as it does not depend on the room geometry at all.

When the L2 model is asked to select a new location it generates a probability value for each location using each of the three distributions. For example it checks how far away the location is ranked in comparison to the furthest ranked location and looks up that rank in the distance distribution. When it has a probability value for each tile from each distribution it creates a final distribution by taking the product of the three values. The tile selection is made by drawing randomly using the final distribution as weights.

The M2 model trains on the human data by creating a library of sample human movements. When M2 is asked to create a path from a to b it rotates and translates each path in the library. It then takes all the paths that stay inside the environment and gives each a rank based on how well it fits. The goodness of fit is determined by the amount the path has to stretch and the change in angle between the stopping angle of the last path segment and the starting angle of this new path segment. M2 then stochastically picks a path from the top ranked paths. The larger the library of paths is the more fluid the path transitions will appear.

An environment novel to an agent is an environment that the agent has never seen, and in particular not trained on. In order for an agent to be generative it must be able to perform in novel environments. The reason generative agents are desirable is that there are many environments and there is not enough time to train an agent on all of them. Some environ-
ments are even created dynamically so the agents behavior may have to be decided at run time. All of these models are generative, because they can all still perform in novel environments. L1 and M1 do not have any training data so in a sense every environment is novel to them. The generative aspects of the M2 and L2 models have been explained. In particular the L2 model uses relative measures in its distributions instead of absolute measures, and the M2 model is better able to generate paths based on the size of its library with the spline model as a fall back if none fit. Therefore all of the agents discussed are generative, and provided with a traversal map and a list of selection locations can perform in a new environment.

**Judging.** For this part of the study we ran 96 human participants recruited from a first-year course in psychology. Each participant was briefed with a description of the task and was asked to judge videos of humans in 4 trials: hiding and seeking in each of the two rooms. Each trial consisted of 5 training videos followed by 10 test videos. The judges knew that the training videos were of human behavior. For the test videos the judges were told that some videos may be of AI agents and some may be of humans but no particular proportion was given. Each video is played at double speed and was between 3 and 44 seconds long. At the end of each test video the judge would rate it as “Human” or “Computer” which would queue in the next video. The judges were also able to re-watch the last shown video.

![Figure 3](image.png)

**Figure 3:** 95% significance bars for agent identification rates. Seek task is in white and Hide task is in red/grey.

**Study Results**

The average identification rate of the judges is shown in Figure 3. The seeking task is shown in white and the hiding task is shown in red/grey. The boxes indicate 95% confidence intervals. The intervals displayed are the Wilson score intervals. The Wilson score interval is used to find a confidence interval on a binomial random variable when there is a chance of the mean being close to the boundaries (0% or 100%). We decided to use the Wilson score interval over the more common normal approximation interval because it is possible one of the agents could have a very high identification rate (near 100%). The closer an agent is to 50% the closer it is to passing the Turing test. If one confidence bar occurs completely under another bar we can safely conclude that the first is identified correctly less than the second with greater than 95% confidence.

In the hide case (red bars), we can see that the M2 model is identified significantly less than the M1 model. This is because the confidence bars for A3 and A4 are completely under the bars for A1 and A2. It is interesting to note that the L2 model did not perform better than the L1 model. We attribute this to there only being 3 selections required in the hide case.

In the seek case (white bars), A4 is still identified significantly less, however A3 is not. This implies that both the L2 and M2 models are required to pass the Turing test for the seek case.

Demonstrating we have 95% confidence that A4 is identified less than A1 is good, but we can find a better confidence. To do this we obtained the scores out of a possible 20 for each judge in each category. This score can be approximated as a normally distributed random variable since it is the sum of individual Bernouli trials. We found, by using a Student’s t-test, a t value of 3.563 in the hide case and 2.723 in the seek case. The degrees of freedom we used was 44 in both the hide and seek cases (24 participants judged A1 and 22 judged A1). Looking these up in a t-table we find a confidence of 99.9992% for hide and 99.08% for seek. From this we can conclude that A4 is indeed identified significantly less than A1. A distribution of the judge scores for agent A1 and agent A4 can be found in Figure 4.

**Discussion**

We attribute the increase in performance of agents A3 and A4 over A1 and A2 in the hide task to using the L2 model since that is the only difference between the agents. The L2 strategy seemingly had no impact in the hide case, however there was a decrease in identification rate between A3 and A4 in the seek case. This implies that the weak simple location selection model (L1) is good enough to pass the hide task, but both of the advanced models, L2 and M2 are needed to pass the seek task. We attribute this to there only being three location selections in the hiding task. The location selection model plays a much smaller part in the hiding task because 3 data points is often not enough to draw an informed guess to the identity of the agent.

Our results indicate that the task we have accomplished is non trivial. Agent A4 performs significantly better than Agent A1 in both hiding and seeking. The mean near 50% in both hiding and seeking indicates agent A4 has passed the conventional definition of the Turing test. That is to say, judges do no better than chance against agent A4.

We can go even further in saying that judges can do no better than chance against agent A4. It is possible, although unlikely, that in the general Turing test a bimodal distribution in judge scores can appear. This would lead to average judge score of 50%, and an incorrect inference that an agent is indistinguishable. For example lets say all human videos start with one turn clockwise and all computer videos start with one turn counterclockwise, but are otherwise indiscernible. If this is the case, every judge can trivially sort
the two types of videos into two bins, clockwise and counter clockwise. However, they will not be sure which bin to label human and which to label agent. This will result in half of the judges making the correct guess and identifying 100% of videos correctly and the other half making the wrong guess and getting 0% correct. This creates a very bimodal distribution in which the mean is 50%, but the agent is very distinguishable. Any judge which is trained by being told agents start clockwise will be able to tell the agents every time.

If all the judges were equal and none could tell the difference all the scores would be left purely up to chance. This means some would score higher than 10 out of 20, some lower, but most would be centered around 10. The shape of the purely chance distribution is included as the baseline in Figure 4 for comparison. The closer a distribution is to the baseline the closer it is to matching pure chance. When we look at figure 4 we can see that agent A4 is normally distributed about a mean of 50%, and in fact is close to the baseline. If we look at the distribution for agent A1, we can see the distribution is closer to a uniform shape than it is to a normal shape. The probability mass near the 100% side of the distribution for agent A1 shows that there exist judges that can consistently identify agent A1 correctly. It is the absence of these expert judges in the distribution of agent A4 that indicates agent A4 really is a strong model and did not show any tells that might consistently give it away to a few very observant people.

**Future Work and Conclusion**

Hiding and seeking is viewed as a fundamental cognitive ability of humans and animals and has several applications in video game design. This paper made the following contributions. We proposed a first computational generative model of hide and seek behavior in adult humans. The model is built automatically by data-mining observed human behavior. We implemented a model within an AI agent and demonstrated its validity via a restricted version of the Turing test. We showed one model was statistically significantly better than a weak model, centered about a mean of 50% and passed a more strict desire for the Turing test of having a normal distribution in its judge scores.

Future work will investigate biasing effects of agent behavior on judge’s perception of human hide/seek behavior. It will also be of interest to incorporate our model into a combat agent in an on-line game such as *Counter-strike: Source*.

**References**


