

Research in Human-level AI using Computer Games

John E. Laird

The goal of our research is to understand what is required for human-level artificial intelligence (AI). A key component of our methodology is developing AI systems in complex, dynamic environments that have many of the properties of the world we inhabit. Although robotics might seem an obvious choice, research in robotics requires solving many difficult problems related to low-level sensing and acting in the real world that are far removed from the cognitive aspects of intelligence. Simulated virtual environments make it possible to by-pass many of these problems, while preserving the need for intelligent real-time decision-making and interaction. Unfortunately, development of realistic virtual environments is an expensive and time-consuming enterprise onto itself and requires expertise in many areas far a field from AI. However, computer games provide us with a source of cheap, reliable, and flexible technology for developing our own virtual environments for research.

Over the last four years, we have been pursuing our research in human-level AI using a variety of computer game engines: Descent 3, Quake II, and Unreal Tournament. Outrage Entertainment, the developer of Descent 3, created an interface for us to test the viability of using a mature AI engine to control a character in the game. Descent 3 is a fun and challenging game that involves three-dimensional control of a spaceship through tunnels and caves. Although it was a useful first step, we abandoned it for Quake II in which the AI system could control more human-like characters. In Quake II, players (including AI “bots”) attempt to shoot each other and they can collect “powerups” such as health items, ammunition, and weapons. Quake II has a dynamically linked library (DLL) that allows access to Quake II’s internal data structures and controls for the computer-controlled bots. We interface our AI engine (Soar) through the DLL to control a bot that a human plays against. One attractive feature of Quake II is that there are editors available to create your own game environments.

Our goal in using Quake II was to discover what was necessary to create an AI bot that played the game in much the same way a human plays the game. We designed our bots to use sensory information similar to that which is available to a human, use the controls similar to those used by a human, and use some of the tactics that humans use. For sensing, the bots can “see” other players and items that are not obstructed by other entities or features (such as walls) in the environment. However, it is difficult to extract spatial information about the physical environment from the game, such as walls and doors, which in the game’s internal data structures are just sets of polygons. The bot needs this information to avoid moving into walls and to create internal maps of its environment. To overcome this difficulty, the bots get range information to the nearest polygons to the front, back, and to both sides. The bots then build up a map as it explores the level that it later uses for moving from room to room, finding the best path to pick up a given powerup, or hiding in corners to surprise the enemy. The bot can also “hear” noises made by other nearby characters. For movement, the bots can move left, right, forward, and back, as well as turn, using commands that map directly onto the actions humans can make by moving their mouse and pressing keys on their keyboard.

The reasoning in our bot is done by programs written in the Soar AI architecture. Programs in Soar consists of sets of rules that support knowledge-rich reactive and goal-driven behavior

through the elaboration of the situation, and the proposal, selection, and application of operators. For example, rules can elaborate the internal representation of the current situation, such as detecting that the bot is too close to a wall, or that a useful weapon is nearby. Proposal rules test the current situation, including elaborations to suggest either primitive or complex operators to perform, such as proposing to pickup a nearby powerup (weapon, health, or ammunition item). Additional rules select among proposed operators, such as preferring to pickup the best powerup if there are multiple powerups nearby. Finally, application rules generate the actions that are involved in performing the operator such as sending a motor command to move forward, or turn.

Many operators that are proposed and selected cannot be applied directly, such as picking up a weapon. These are automatically converted into subgoals where further rules propose finer-grain operators to achieve the more abstract operators. Figure 2 shows a small part of the hierarchy that can arise as part of exploring a level. [Add Figure] We did some informal studies where we had humans compare the behavior of human players to variations of the bots to determine if changes in decision speed, tactics, aggressiveness, and aiming skill influence how “human” the bots were. The trends in the results were that bots with extremely accurate aiming or extremely fast (< 25 msec.) decision speed appeared less human than ones with less accurate aiming skills and slower (100 msec.) decision speed.

We also did a qualitative analysis of the behavior and noticed that expert players attempt to anticipate the actions of their opponents. Anticipation is a form of planning similar to the look-ahead search performed by AI programs that play classic games like chess and checkers; however, the challenge in a game like Quake II is that when a decision is made to perform an action (such as when to turn) is often as important as the choice of action to take. Moreover, in contrast to chess where you can see the complete board, in Quake II there is only imperfect information about the state of each player. To simplify the process, our bot creates an internal representation of what it thinks the opponent’s internal state is, and then uses its own tactics to predict the opponent’s behavior. It continues to predict until it finds a situation in which it can get to one of the opponent’s destinations first and set an ambush, or there is so much uncertainty in what the opponent will do that it is not worth projecting its behavior any further. After adding anticipation, playing the bot shifts from being a purely tactical game of trying to get the best weapons and shoot the fastest, to a more strategic and intriguing game, where you are always wondering if the bot has already second guessed you and is hiding in ambush on the other side of the next door.

Although action games such as Quake is the most popular game genre, there are inherent limits in the complexity of behaviors required to create compelling bots that are essentially computerized punching bags. Furthermore, these games limit the human gaming experience to violent interactions with other humans and bots. Therefore, we are currently working to develop non-violent plot-driven computer games where the AI characters have diverse and complex behavior that is driven by the interaction of their body with the environment, their goals, their knowledge of the world they inhabit, their own personal history, and their interactions with human players. This will lead to games where the human players are faced with challenges and obstacles that require meaningful interactions with the AI characters. We are building on one of the oldest genres of computer games, sometimes called interactive fiction or adventure games, which involve having the human player overcome obstacles and solve puzzles in pursuit of some

goal. [Myst, Bladerunner, Monkey Island series] One weakness of these games is that the behavior of non-player AI characters is scripted, so that the interactions with them are very stilted and not very compelling. Our challenge will be to create AI characters whose behavior are not only human-like but also leads to engaging game play.

Using Unreal Tournament (UT), we are creating an adventure game where the player takes on the persona of a ghost-like energy creature trapped in a house. UT is an action game similar to Quake2 with an underlying engine that is extremely flexible. For just the cost of the game (\$20), you get access to level editors for defining the environment, a scripting language (Unrealscript) for defining the “physics” of the world and the way objects in the world interact, and the ability to import your own objects into the game.

In our game, the human player’s goal as the “ghost” is to escape the house and return home to an underground cavern. The ghost is severely limited in its ability to manipulate the environment. It can move or pick up light objects, such as a match or a piece of paper, but it can’t move or manipulate heavy objects. Moreover, metal drains the ghost energy, so the ghost must avoid metal objects. These constraints force the player to entice, cajole, threaten, or frighten the AI characters into manipulating the objects in the world, which in turn forces us to develop AI characters that have enough “intelligence” to make these social manipulations possible and realistic. With the AI characters playing such a central role, they must have distinctive personalities in terms of their goals and reaction to the environment. For example, there will be the evil scientist who is immune to fear but is weak and easily fatigued by exertion or cold and wants to capture the ghost character, while there will also be a lost hitchhiker (we aren’t trying to have the most original story ever) who is easily frightened by the ghost, but is physically strong and driven by curiosity. The game will push our research to integrate the knowledge-based, goal-oriented reasoning that we have concentrated in the past, with emotions, personality, and physical drives that have been used in simple, knowledge-lean agents in other systems [references to Oz, Sims]. Our hope is that we inspire others to pursue human-level AI characters and new types of games that those characters make possible.

References (to be completed)...

Figures: I can get some screen shots of our UT game.