1

Game AI Appreciation, Revisited

Mike Lewis and Kevin Dill

1.1 Introduction

This book represents a collection of knowledge, advice, hard-learned wisdom, and insight gathered from across the community of developers and researchers who have taken an interest in game AI. With such a formidable lineup of expertise brought to bear on the subject, it may come as a surprise to learn that even the leading experts in the field of game AI find it challenging to define just what exactly that field comprises.

Concocting a reasonably concise definition of game AI is hard enough—let alone reaching consensus on that definition with the rest of the development community. Often, the question is evaded entirely. AI can be recognized when it is observed in action, so why bother trying to hem it in with a strict definition?

Unfortunately, this perspective does a fundamental disservice to the rest of the game community. Players, journalists, critics, and even other developers are often left with an incomplete idea of what exactly game AI is all about, what it can offer, and what is (and isn't) easy. A comprehensive analysis could fill a book by itself, but we will at least attempt to sketch out the boundaries of game AI here.

Music, classical art forms, film, theater, and even food all benefit from the idea of "appreciation courses." Without some foundational knowledge of how songs are composed or sculptures revealed within a block of marble, it can be difficult to truly grasp

and enjoy what makes them so special. In this chapter, we will attempt to explore the field of game AI from a similar perspective.

1.2 What Is Game AI?

One of the first things established in any appreciation course is *vocabulary*. A shared lexicon makes communication much more fluid and much less error prone. Given that this is a book about game AI, it seems only natural to begin by defining the very term "game AI."

Although even the term "game" is notoriously difficult to pin down, we will assume that it is generally understood well enough. We are particularly interested in electronic computer games in that they are the most readily visible dwelling place for contemporary game AI. The important aspects of games that we consider in this chapter consist primarily of the gameplayer's *experience* as presented through the constraints of a *design*.

The design of a game is, loosely speaking, the setting and rules of the artificial "world" in which the game takes place—however abstract, fantastical, or realistic that world may be. A *designer* manages and sculpts the vision for how the game will operate, how the player is expected to interact with the game, and what sorts of experiences the player should be having while playing.

Typical components of the experience are things the player might observe (such as opponents, environments, and narrative moments), methods for interacting with the game itself, and *feedback* that provides information to the player about the consequences of those interactions.

Creating a game requires the use of a diverse array of tools, ranging from the purely artistic to the intensely technical. All of these tools are involved in enabling the vision of the game's design. AI is a particularly powerful part of that toolbox.

If game AI is a toolset, then when do those tools become necessary? Games can be (and occasionally are) created with no real "artificial intelligence" component. Clearly, it is not necessary to have particularly sophisticated AI to deliver an experience like Tetris. So when do these vision-enabling tools actually get deployed?

A key differentiator between games and some of the more common classical art forms is that games are inherently *interactive*. The player has some role, whether minor or central, in shaping the experience that unfolds during play. In more traditional arenas, participatory theater (such as improvisational comedy) invites the audience into an exchange with the performers. For games, there isn't necessarily a ready cast of human performers with whom to interact—and yet we still consider games *interactive*. Within this notion of interactivity lies an important signpost on the route to defining game AI.

Interaction can take on many forms. Strictly speaking, people interact with inanimate objects all the time. But interacting with other *people* is generally much more interesting than, say, turning on a light switch. While game experiences can be almost entirely mechanical and "inanimate" in a sense, many games seek to deliver something more compelling—something that seems more tangible, more relatable, more *real*.

We may be able to relate to characters or situations in a film or novel, but we can't interact with them. By contrast, our experience of a game is funneled through our interactions with the game universe. It is the sensation that something on the other side of the screen is

almost *alive* that makes game AI unique and powerful. This feeling arises from observing the game "making decisions" about what happens next.

More broadly, game AI can be thought of as applied decision making, specifically in the context of games. Most often, this takes the form of an autonomous, decision-making agent (i.e., an NPC) in a virtual world. Game AI need not be limited to just these sorts of applications. For game AI practitioners, the goal is to use programmatic techniques to enable the computer to make effective decisions that support the experience they aim to create. These decisions might entail almost anything, from characters and their split-second reactions to in-game events, to sweeping changes to an entire set of game rules, based on models suggested by various algorithms [Schwab 14].

1.3 Fuzzy Border

Artificial intelligence as a field is expansive and encompasses quite a bit more than just games. It is worth noting that much of what is categorized as "AI" in nongaming spheres bears little resemblance to the typical technology used in games.

For example, academic AI commonly focuses on building a falsifiable scientific model of some cognitive processes. Popular reasoning packages from outside the game's domain, such as SOAR, ACT-R, and CoJACK, often take this approach. Changing the behavior of the AI requires refining and adjusting the cognitive model itself. While this approach is important for drawing sound conclusions from research and making verifiable predictions, it introduces a layer of abstraction between the game designer's intent and the AI's implementation. This added abstraction increases the required amount of development effort, because changes to the AI behavior cannot be made directly. Moreover, this abstraction introduces a significant computational cost, which can be prohibitive in many gaming applications.

For a game, we want to achieve a particular behavior to suit the design goals; how that behavior is arrived at is more or less inconsequential. More importantly, we want a simple set of controls with understandable and predictable effects, so that creating the desired experience is straightforward. Excess abstraction and complexity often serve to make it more difficult to attain the precise game experience that we want.

Another area of AI with considerable importance outside of games is machine learning. Allowing a computer to discover and tune its own solutions may sound appealing, but this approach frequently conflicts with the desire to create a specific sort of experience. Machine learning has been successfully applied in games, but mostly in limited subdomains, such as tuning paths taken by cars in racing games [Manslow 01]. Other uses include actual game design mechanics, which can be immensely entertaining and yet equally difficult to test [Barnes 02]. More recently, learning has been explored as a method for calibrating game balance [Tozour 13]. All too often, though, it poses a challenge that games simply don't need to address.

In a similar vein, many AI techniques (such as planning and utility-based reasoning) are designed to produce a high degree of *autonomy*. This characteristic enables AI agents to make sensible decisions in circumstances that were not directly anticipated by the developers. While autonomy in an AI agent can often be desirable, if it is not carefully constrained, it can lead to unpredictable behavior (sometimes referred to as "emergent behavior"). Whether or not this should be considered a positive outcome is hotly debated

within the community, but ultimately, it depends highly on the game design and the nature of the experience we wish to create.

Authorial control and autonomy are fundamentally at odds, and it is up to each individual game designer to decide which side to favor and how strongly. Some games, like *The Sims*, thrive on autonomy—each individual character has almost complete freedom to act and even "live" within the game world. Some entire genres, such as real-time or turn-based strategy games, require a high a degree of autonomous behavior in order to deliver compelling opponents. On the other end of the spectrum, games like *World of Warcraft* have deliberately chosen to minimize autonomy. The AI in a typical *World of Warcraft* boss fight is very carefully hand-tuned to deliver a specific experience, but it is also extremely predictable.

One more area of burgeoning AI research is data mining. It is as yet unclear how much this field will overlap with game AI in the future. For the time being, though, there is only minimal intersection. In a few noteworthy cases, data mining is used to help calibrate and balance game designs (whether before or after shipping). However, the interaction between this data analysis and the resulting functionality of the game AI typically involves a significant element of human intervention. Much like with machine learning, there is a fundamental tension with the desire for authorial control, but still some significant potential.

Since the goals of game AI are so frequently different than those of other areas of AI research and application, the selection of techniques that appear in games naturally differs as well. As a general trend, game AI solutions favor simplicity, ease of tuning, and (to an extent, depending on the game) the ability to precisely control the resultant behaviors. While there can be considerable overlap with AI techniques from other fields, games are quite simply solving a different set of problems.

1.4 AI as the Nexus of Game Development

Game AI and game design are virtually inseparable—and indeed, many teams lean heavily upon hybrid design/engineering roles or even unify the "lead design" and "lead AI" roles entirely. As established previously, the entire purpose of game AI is to further the creation of compelling experiences. The nature and scope of those experiences depends entirely on the design vision of the game itself. Any healthy game development team will exhibit close interaction between design and programming, whether the roles are combined or not.

This relationship is no coincidence. For a large number of games, the design is itself expressed to the player through the medium of the AI. In order to communicate the nature of the game world and the agents within it, designers rely on AI, which in turn is brought to life in large part by programming efforts. Refining the behavior requires returning the focus to design, and the feedback and iteration loop perpetuates in this way.

Beyond just design, AI has ties to almost every part of the game development process. Models, sprites, textures, effects, animations, and other art assets define the visual language that AI agents can use to communicate with the player. Audio enables direct verbal interaction with the player. Even music can be dynamically selected and modified at run time in order to evoke specific reactions from the player.

Animation and AI exhibit a very similar relationship as AI and design. Much as the game design is largely expressed via the AI, the AI itself is largely expressed via animation. Animation is often the primary language by which the AI communicates its actions, its intentions, and its opinions and feelings to the player. The AI will only be perceived to be as good as the animations it uses. By the same token, all the fancy animations in the world are useless if the AI never chooses to use them or uses them poorly or inappropriately.

Another common tool for communication with the player is audio and particularly voice. Again, this is a case where the effectiveness of the audio is deeply tied to the power of the AI, and the perception of believability in the AI can be radically affected by careful voice work [Redding 09].

Physics systems are often vital to AI implementation, although the reverse is not true. That is, physics can provide information about the world (such as through raycasts) that the AI relies on to drive its decision-making processes. On the other side of the sense–think–act pipeline, physics often constrains the ways in which agents can interact with the game world. For instance, setting an actor's position by directly modifying its transform might violate the integrity of the physics simulation, so it becomes necessary to use the physical simulation to move that actor into place. This interplay can often lead to interesting problems, such as determining how to move an agent through the world to achieve a certain goal. As the complexity of these challenges grows, it becomes increasingly necessary to seek out novel solutions [Sunshine-Hill 14].

Even systems that are largely independent of the AI, such as rendering, can often be utilized by the AI to help portray intent. A common technique is the use of "tells"—graphical effects or very simple animations that are used to foreshadow upcoming actions on the part of the AI [Abercrombie 14]. One area that seems to have been less thoroughly explored is the interaction between AI and user interfaces. AI pathfinding has occasionally been used to show routing hints to the player. However, many more possibilities exist. For instance, UI elements could be selectively hidden or shown based on signals from the AI systems. Any UI offering a list of targets or objectives could be sorted or filtered based on intelligent criteria—deploying AI reasoning not necessarily in service of the computer's agents, but instead for the benefit of the human player.

For online games, AI can represent an interesting technical challenge. There are rarely one-size-fits-all solutions for replicating AI behavior on multiple distinct devices over a network. Making the AI behave in a consistent and timely manner on all relevant devices, while simultaneously trying to minimize both bandwidth and computational requirements, typically requires a considerable amount of effort. Even though bandwidth has increased dramatically since the early days of online gaming, a few bytes saved here and there can make a significant difference for games on cellular networks or with massive numbers of players.

Beyond the realm of what the player directly experiences, there is considerable potential for the use of AI in tools—both for developing games preship and for analyzing (and perhaps updating) them postrelease. While tools for debugging and observing AI are widely recognized as vital to creating sophisticated behaviors, it is far less common to see tools directly powered by (or even heavily supplemented by) AI systems. Notable exceptions do exist, and as data mining techniques are refined, it seems highly probable that deeply AI-bound tools will become more and more prevalent.

1.5 Frontiers of the Field

AI in games has, as a whole, been improving appreciably over time. What began as trivial pattern-based movement has evolved into spatial awareness and tactical reasoning. Opponents that once were limited to a single means of antagonizing the player can now cleverly take advantage of elements of the game world to invent new challenges on the fly. Even the presentation of an AI character's activities has melded into sophisticated animation and audio systems.

With all of this progress and with resources like this book filled with ever more powerful ideas and techniques, it may be tempting to assume that game AI has conquered the "easy stuff" already. And while some challenges are indeed understood well enough to have de facto standard solutions, it doesn't take much wandering off the beaten path to discover that the field of game AI contains very few truly "solved" problems. Even challenges with powerful theoretical solutions often become vastly more difficult when thrust into the ever more complex environments of contemporary games.

1.5.1 Pathfinding

Implementing a computer game featuring moving agents typically involves finding paths from one place to another through a (potentially complex) world space. Fortunately, efficient solutions to basic forms of the problem have existed since 1959 [Dijkstra 59], with notable improvements (in the form of A*) appearing less than a decade later [Hart 68].

The popularization of A* search has led to the widespread misconception that finding paths is a nonissue in modern games. It is certainly true that this algorithm and several noteworthy variants have had tremendous success in practical applications and will almost certainly remain a staple of pathfinding solutions for a long time to come. However, many games make concessions that enable A* to be particularly effective.

The spatial representation used by the game is a common area for such concessions. Navmeshes are almost ubiquitous in 3D games now, but they quickly begin to struggle if, say, the environment can be radically altered during gameplay in unpredictable ways. Solutions exist, but are not well understood or publicized, and are challenging to implement. (A large portion of them exist in middleware path-planning solutions and are regarded as trade secrets.) Even a static navmesh is fundamentally an approximation of space. It is not possible to guarantee shortest "real" paths when searching an abstract representation of approximate space.

On that note, most pathfinding techniques are only concerned with optimality in terms of distance and/or time; in other words, they are interested in finding the shortest or quickest route from point to point.

What if we want to have a group of agents split into several squads and follow the *four* "best" paths to a destination? What if we want one squad to pin the enemy in place, while another flanks them? What if agents want to find a path that minimizes their visibility to the enemy? What if the player can choose to place (or remove!) walkable areas at a whim? What if significant portions of the decision will change over time, such as when an agent wishes to use a slow-moving vehicle for temporary cover? What if the agent is an easily distracted puppy and prefers to bound to the left and the right as it walks?

Going further afield, what if an agent has many different ways to move around, such as jumping, flying, and swimming? How do all of those options get considered effectively?

How do we decide when to vault over an obstacle and when to simply go around? What if the agent is moving in three dimensions, such as a bird or dolphin? How do we decide where to place the agent vertically and when to change that position? What happens when the path space is crowded by other agents moving about? What if the path is meant for use by a snake (or spider, or elephant, or 18-wheeled truck) instead of a bipedal humanoid? How do we make the motion in these scenarios look convincing?

These are less well-trodden problems. Many game engines address at least some of them, and there are middleware vendors developing ever-increasingly impressive solutions. Yet many of the details are highly game specific, or even completely unsolved. As the sophistication of path requirements increases, so too does the likelihood of running into a truly difficult challenge.

When you play a game with AI characters that can move freely in the simulated world, try looking for cases where the AI favors "shortest" paths over more realistic or genuinely optimal paths. The AI characters taking such routes often lose all credibility. Players still spot completely pathological cases such as agents getting stuck against walls or running into each other in the wild. Watching game AI agents navigate their surroundings with a critical eye turns up plenty of opportunities for improvement.

Finding the shortest path is very close to a solved problem (in cases where the environment is sufficiently simple or amenable to an accurate search heuristic), but finding truly *great* paths remains an area of ongoing exploration.

1.5.2 Conversations

Games that offer the player the opportunity to engage in conversation with AI characters have traditionally encountered a series of nasty obstacles. Typically, game conversations are either purely scripted (e.g., the player may watch a noninteractive "cutscene" of the conversation) or, at best, powered by simple dialogue trees. This is almost entirely due to the difficulty of making genuine conversational interactions possible with an AI agent.

Dialogue trees remain popular but suffer from a geometric increase in the amount of necessary content as the degree of branching goes up. The more possibilities the conversation can explore and the longer the conversation might last, the more text must be written (and, perhaps, audio recorded) to cover all the options. As such, most dialogue trees stay quite limited and often seem artificially constrained to the player.

Developing better conversational interactions with AI quickly encroaches on natural language processing territory. Eventually, it becomes impractical to hand-author (and possibly record) all the dialogues possible. The extreme option is to parse the player's input on the fly and generate believable responses dynamically as well.

Tempting as this may sound, natural language processing is widely regarded as a highly difficult AI problem. While great inroads have been made in the area over the past decades, the natural language field as a whole still offers plenty of interesting challenges to tackle.

Intuitively, it seems that there must be some intermediate solutions between canned dialogue and full natural language conversation. While some approaches are available already, they typically rely on an enormous corpus of knowledge to operate—something that may not be practical to include in a game. Moreover, this knowledge base often comes from real-world data such as Internet resources, making it useless for an AI that is supposed to exist in, say, medieval Europe.

Some highly noteworthy exploration into conversations for games exists, and the results hold promise and excitement for anyone interested in better human/AI interactions [Evans 12, Orkin 07, Mateas 03, McCoy 13]. Even still, there is plenty of fertile ground for further investigation.

Any player who has complained about dialogue trees and their limited selection has felt firsthand the constraints of typical human/AI conversations. Even widely celebrated "chat bots" in the spirit of *ELIZA* [Weizenbaum 66] can be confounded or otherwise reveal their artificiality within the confines of relatively normal interactions.

Try listening to the conversations between "your" player character and other AI characters in a contemporary game. Every time something important cannot be said, every time something totally "wrong" is available as a choice, and every time you get bored of hearing the exact same dialogue dozens of times, think about the state of the art in human/AI interaction and where it might be pushed further.

1.5.3 Dynamic Storylines

As the cost of developing high-quality games continues to increase, it is more important than ever to maximize the value of every ounce of effort invested in a game's creation. The tension between the finite budget for developing a title and the amount of content the players expect from the final product requires careful balance. Budgets can only grow so large before it is simply impossible for a title to be profitable at the price points expected by consumers.

One way to address this problem is via *replayability*—if a title can be enjoyed multiple times without growing stale, the value to the player is increased, while the cost of producing that game remains fixed. Games like Tetris or chess are infinitely replayable, whereas a heavily narrative-driven, "cinematic" experience might only be worth playing through once. But what if we want to make a game that has a rich storytelling aspect *and* can be replayed many times?

There are many challenges in building a highly branching narrative with numerous opportunities for things to play out differently. In a sense, the problem closely resembles a more generalized version of the issue with dialogue trees. This is where dynamic storytelling enters the scene. Instead of hand-authoring all of the possible narrative states of the game, we use some AI to take on the role of storyteller.

The storyteller must manage pacing, tension, difficulty level, plot points, and so on, while simultaneously reacting in a believable and entertaining way to the player's decisions. This may even wind up feeding into the knowledge representation used by a dynamic conversation system, so that the player can converse with characters in the game world about what has been going on.

A crucial element of interactive storytelling is guiding the narrative toward critical moments and ultimately to a conclusion. This is difficult enough even without considering players who attempt to deliberately derail the story. As the players' freedom to interact with the game world increases, so do the number of possible outcomes of their actions. A major challenge with dynamic storytelling is to produce this guided narrative without feeling forced or jarring.

Steps have certainly been taken in this vein, such as the AI director in *Left 4 Dead* [Newell 08]. Even classic "choose your own adventure" style narratives can be considered a prototypical form of dynamic storytelling. There are two contrasting dynamic storytelling

approaches represented here. For "choose your own adventure," the player selects a path through a preconfigured set of possible storylines. There is a strong sense in which the player is being told a story. On the flip side, AI director style systems focus on generating an interesting experience for the player to have; the story is emergent and may even only really become apparent once the player recounts that experience.

Both methods are completely valid, although the role of AI systems can be markedly different. (Indeed, the "choose your own adventure" book format predates sophisticated game AI.) In any event, there is a substantial amount of possibility in the field of dynamic storytelling, and the ultimate role of AI as storyteller is still being defined.

For anyone who has played a well-run pen and paper role-playing game, or even experienced a rousing session of "Let's Pretend," it isn't hard to find storytelling deficiencies in computer games. Eventually, someone playing is bound to try something that wasn't specifically accounted for in the game's construction. While this can lead to entertaining (albeit somewhat absurd) results, it more often leads to mild annoyance and disappointment.

When playing a highly narrative-driven game, take time to observe the ways in which player choices are handled. Note when choices are made available and the range of options provided. Try doing something that subtly (or wildly) clashes with the sorts of choices you are expected to be making. Anything that just doesn't quite "work" is likely to be a sign of the fringes of where computer-guided narrative has been taken. See if you can think of ways the developer could (practically and cost-effectively) extend the freedom of the player and the replayability of the game, without compromising narrative integrity.

1.5.4 Player Modeling

Software developers from across the tech sector often lament that we cannot intuit the true intentions of the player (or, more generally, the user). The available mechanisms for human/computer interaction are fundamentally limited. It should come as no real surprise that the full thought process of the human mind loses something in translation when crammed through the funnel of pushing keys and clicking mouse buttons.

A tremendous amount of effort goes into developing user experiences and researching how to streamline them. Most game studios are familiar with the routine of *usability testing*, where outside players are invited to play a game in development. By observing these players, the developer can glean huge amounts of valuable insight into where the game's user experience shines and where it needs improvement.

Generally, the feedback obtained from usability testing is only available during the development cycle of the game itself. Once the game ships, it is no longer viable to put eye-tracking hardware on each player or rig them up with electrodes. Outside of controlled, laboratory-style settings, we have a limited toolset for understanding how our players are thinking and how they are likely to behave.

Thankfully, the trappings of a high-tech behavioral science lab are not mandatory for analyzing player behavior. One increasingly popular technique is to harvest statistics from thousands or millions of players, collect them automatically via the Internet, and then comb through them for interesting revelations. Online games have been doing this extensively for years, although there is no strict reason why it could not be done for any game with access to a network connection.

Ultimately, the dream of collecting all these data is to build a *player model*. This is a statistical description of how players are likely to behave, often separating them into major

groupings based on their play style. If this model is sufficiently accurate, it becomes possible not just to study player actions after the fact but also to predict *future* actions.

Once we have a mechanism for predicting player behavior, we can start customizing the game experience for each individual player based on the model. If a particular player is known to enjoy exploring large areas of the game world and collecting rare items, for example, the game can dynamically highlight similar opportunities for the player, or even generate new ones on the fly.

Much of this is already possible from a technical standpoint. In fact, games such as *League of Legends* exploit player modeling for, among other things, designing new content updates, which directly drive revenue streams [Lin 14]. However, the process is often long, arduous, and highly error prone.

Ideally, player modeling should be able to customize a game as it is being played, with response times measured in seconds—if not faster. Accomplishing this may seem like a predominantly mathematical task, but AI techniques for heuristically solving complex modeling problems are intensely promising.

Statistical modeling is still far more art than science and requires a substantial degree of expertise to apply effectively. Moreover, the challenges of gathering and processing millions of data points—sometimes millions *per day*—should not be underestimated. Turnaround time for player modeling is often measured in days—a far cry from the fractions of a second available during live gameplay.

While monetization opportunities are often the focus of research in this realm, there is plenty of fodder for improving the game experience itself. It can be tempting to limit our exploration of player modeling to simply trying to predict what a player wants or will do. A much more intriguing option is predicting what the player does not expect but will still be pleasantly surprised to discover.

Find any group of players discussing a particular game, and you're likely to find divergent opinions on the nature of the game experience itself. What if games could tailor themselves specifically to their individual audiences and deliver an excellent experience for everyone? One of the trickiest problems in game design is appealing to a broad enough audience to be financially successful without overtly diluting the purpose of the game itself. AI has the potential to significantly impact that balance.

1.5.5 Modeling and Displaying Emotion

The past several years have seen animation becoming an increasingly important element of game AI's presentation to the player. Without high-fidelity movement, it can be difficult to get AI agents to appear convincing and intelligent. While much progress has been made in animating humanoid bodies, there is still a lot of work to be done on the more subtle and powerful portrayal of emotions.

As early as the 1940s, it was noted that people tended to attribute emotions to animated shapes based purely on their spatial relationships and motions [Heider 44]. Until fairly recently, very few games could provide the visual fidelity required to do much more than this in terms of conveying emotion and intent.

Animation technology is already capable of incredible feats when it comes to facial details and expressions. Building a sufficient model of emotional state is also clearly well within reach, as *The Sims* demonstrated. The major source of difficulty lies in the realm

of audio. Without the ability to modulate voices realistically, games often fall back on reusing voice assets in situations where they sound jarringly out of place.

Consider any moment when the immersion of a game experience has been broken by a character seeming stilted, robotic, or otherwise emotionally unrealistic. While carefully sculpted narrative moments and cutscenes can generally portray emotional state fairly well, the bulk of gameplay is often fraught with such moments of surreal absurdity. Modeling and displaying emotional richness will be essential to mitigating this in future games.

Advances in this area will unlock a vast amount of potential for AI characters and liberate game developers from the constraints of the fairly limited emotional palette currently available. Combined with dynamically generated dialogue and AI-assisted storytelling, emotional modeling represents a tremendous opportunity for creating games that are more immersive and convincing than ever before.

1.5.6 Social Relationships

Once AI characters are able to model and display emotions, we quickly run headlong into another difficult problem: what happens when two AI characters interact with each other? Delivering a believable scenario requires some degree of simulated social interaction. Some games have already experimented heavily with this [McCoy 13], but for the most part, relationships between AI characters have historically been rigidly scripted out by designers and writers.

What qualifies as a reasonable model of social behavior in a game might differ substantially from models used in more academic and scientific circles. A game's setting and narrative may only call for a limited number of interactions between AI characters, for instance, permitting the social model to be substantially simplified. On the other end of the spectrum, role-playing games or open-world "sandbox" games might benefit from fairly sophisticated simulations of social interaction.

As with emotions, however, modeling social behavior is not the end of the story. If the player doesn't have a way to observe the model's state and how it evolves during play, it may as well not exist. In fact, since the player has no way to understand why an NPC has made a particular choice, a hidden model can do more harm than good. The results of this social simulation will be most effectively portrayed when a variety of expressive options are available to an AI character. Generating dialogue, displaying emotional status, and selecting relevant animations will all be important parts of selling the social model to players.

1.5.7 Scale

Games are quite a bit different from many other realms of AI research in that they are constrained fairly heavily in terms of computational resources. A game might need to be playable on a cell phone, for instance. Even on more capable hardware, it is not uncommon for the lion's share of the resources to be dedicated to graphics and physics. By contrast, more traditional AI applications may have an entire machine—or network of machines—to themselves. Game AI systems are typically constrained to around ten percent of the available processing power, and even the most AI-intensive games rarely allow more than twenty percent. Moreover, many games require near-real-time interactivity with

the player. The combined limitations on memory, CPU resources, and execution time can be fiendishly difficult to overcome.

Another important challenge for games is that they are typically pushing the performance of the host machine to the limit. Programming techniques that are marginally inefficient (e.g., creation of memory fragmentation, frequent cache misses) can easily add up to prohibitive performance problems, even if isolated systems are well within their defined limits.

The difficulties here lie more in engineering than theoretical AI techniques, although careful selection of techniques has a tremendous impact on the degree of ingenuity required. Understanding the time and memory requirements of various solutions is vital to selecting appropriate technical strategies.

Individual AI agents are becoming increasingly sophisticated. Decision-making techniques are often quite efficient in isolation, but the same is not always true for environmental awareness, perception, searches, dynamic state-space analysis methods, and so on. Gathering enough data to make an *informed* decision, or to execute that decision in a complex environment, can be costly.

Creating a small number of highly believable agents is challenging, but well within reach of modern game AI techniques. However, many games are increasingly vast in geographical terms, and others may call for huge numbers of agents interacting simultaneously. As AI agents proliferate, performance and memory usage become serious concerns.

It may be tempting to conclude that these challenges will eventually dissipate as computing hardware continues to advance. However, the slowing pace of CPU speed improvements and the ongoing difficulties surrounding effective use of massively parallel architectures dampen this hope substantially. Worse, history has shown that more powerful hardware tends to lead to a demand for correspondingly richer game experiences. For example, Damián Isla (AI developer on the *Halo* series) once remarked that *Halo 3* was actually able to do *fewer* line of sight checks than *Halo 2*, despite running on substantially more powerful hardware [Isla 09]. Environments had become so much more detailed and complex that the increased cost of an individual check outstripped the pace of Moore's law.

With these sorts of trade-offs facing developers on a routine basis, it is likely that scale will continue to be a source of interesting problems for quite some time to come.

1.5.8 Content Explosion

In the vein of practical challenges, the sheer quantity of hand-crafted assets and content used by even medium-scale contemporary games can be daunting. To move around the world convincingly, AI agents typically draw on a library of animations; more sophisticated movement requires a corresponding increase in the animation effort necessary to maintain a given level of desired fidelity.

Other types of assets can impose significant limitations on the expressivity of AI as well. If audio recordings are used for verbal communication, an AI character is restricted to "speaking" only the lines that have been recorded. Even without the limitation of voice-over recordings, hand-authored text for dialogue can represent a significant volume of content as well. For an AI agent to interact meaningfully with its surrounding world, the agent needs a selection of props and points of interest, as well as knowledge of how to make use of them. Such content can also proliferate quickly as world complexity increases.

In general, the so-called "content explosion" problem is becoming a substantial obstacle to the continued advance of game realism, fidelity, and richness. Every individual bit of detail that goes into the world leads to a combinatorial increase in the amount of content required to effectively make use of that detail from an AI perspective. Eventually, the cost of creating the content dwarfs the potential benefits of the addition itself, and reality slams headlong into a sort of upper bound on the sophistication of the game world.

Perhaps the most visible approach to addressing this problem (besides just accepting limited scope as a fact of life) lies in the realm of *procedurally generated content*. In essence, a computable algorithm is used to augment (or even supplant) direct human creativity. By relegating some portion of the content generation problem to an automated and repeatable process, the overhead of producing content can be diminished dramatically.

Procedural techniques have been used (with varying degrees of success) in areas such as world or map generation, storyline or "quest" creation, and even the dynamic emission of high-quality dialogue and narrative text. Animation already benefits from procedural augmentation, and research into broadening the applicability of procedural techniques to animation continues. Voice synthesis and recognition technologies advance at a rapid pace, offering the very real possibility of replacing human actors with computer voices in the not-so-distant future.

As an example of the sorts of challenges that procedural generation faces, though, voice synthesis becomes dramatically more difficult as the level of fidelity increases. Computers have been able to make rudimentary utterances for decades, and yet successfully portraying emotional cues—or even generating truly human-sounding voices—remains an area of ongoing research. To underscore the difficulty, the game *Portal* (which included an AI entity as a major character) modified the recordings of a human voice actor to sound synthesized, rather than trying to make a fully synthesized voice sound sufficiently human [Wolpaw 07].

In general, procedural content generation represents a promising but largely undeveloped frontier. While the theoretical benefits are considerable, it remains to be seen how much effort it will take to attain them.

1.5.9 The Unexplored

Beyond the relatively short list of difficulties enumerated here, there lies a vast realm of uncertainty. There is without a doubt a long list of questions that have yet to even be asked. In a very real sense, we have barely begun to scratch the surface of what game AI can do.

New forms of technology are constantly arriving on the scene, and a large portion of them can and will influence the direction of gaming in the future. Wherever these devices and mediums take us, game AI will be sure to follow close behind. For anyone with an interest in game AI, this is an exciting time. Not only do we have the opportunity to leverage a vast amount of existing technical capability and know-how, but we can also look forward excitedly to the games and AI of tomorrow.

1.6 Conclusion

With the proliferation of computer-powered games comes an ever-increasing desire to push the boundaries of the experiences those games can offer. As the seemingly limitless imagination of designers continues to generate fantastic new ideas for players to explore, there is no doubt that AI will play a powerful role in delivering those experiences.

Although many areas of game implementation are fairly well understood, there is still ample room for innovation and discovery. Game AI is no exception to this pattern. Developers have amassed a considerable body of formalized techniques and standard approaches to common problems—but many more challenges remain to be conquered.

The remainder of this book explores a diverse and rich slice of knowledge from contributors throughout the game AI community. The frontiers of game AI have advanced quickly over the past decades, and perhaps one of the most exciting possibilities created by books like this one is that any one of our readers could be instrumental in pushing those boundaries forward for games yet to come.

References

[Abercrombie 14] Abercrombie, J. 2014. Bringing bioShock infinite's Elizabeth to life: An AI development postmortem. *Lecture, Game Developer's Conference AI Summit 2014*, San Francisco, CA.

[Barnes 02] Barnes, J. and Hutchens, J. 2002. Testing undefined behavior as a result of learning. In *AI Game Programming Wisdom*, ed. S. Rabin. Boston, MA: Charles River Media.

[Dijkstra 59] Dijkstra, E.W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1: 269–271.

[Evans 12] Evans, R. and Short, E. 2012. Beyond Eliza: Constructing socially engaging AI. *Lecture, Game Developer's Conference AI Summit 2012,* San Francisco, CA.

[Hart 68] Hart, P.E., Nilsson, N.J., and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics SSC4* 4(2): 100–107.

[Heider 44] Heider, F. and Simmel, M. 1944. An experimental study of apparent behavior. *American Journal of Psychology* 57: 243–259.

[Isla 09] Isla, D. 2009. *Artificial Intelligence Panel*. Boston, MA: Post Mortem Group.

[Lin 14] Lin, J. 2014. *Keynote, New Economic Models and Opportunities for Digital Games*, York, United Kingdom.

[Manslow 01] Manslow, J. 2001. Using a neural network in a game: A concrete example. In *Game Programming Gems*, Vol. 2, ed. M. DeLoura. Boston, MA: Cengage Learning.

[Mateas 03] Mateas, M. and Stern, A. 2003. Façade*:* An experiment in building a fully-realized interactive demo. *Lecture, Game Developer's Conference 2003*, San Jose, CA.

[McCoy 13] McCoy, J., Treanor, M., Samuel, B., Reed, A., Mateas, M., and Wardrip-Fruin, N. 2013. Prom week: Designing past the game/story dilemma. *Proceedings of the Eighth International Conference on the Foundations of Digital Games,* Chania, Crete, Greece.

[Newell 08] Newell, G. 2008. Gabe Newell writes for edge. http://www.edge-online.com/features/gabe-newell-writes-edge/(accessed January 17, 2015).

[Orkin 07] Orkin, J. and Roy, D. 2007. The restaurant game: Learning social behavior and language from thousands of players online. *Journal of Game Development* 3(1): 39–60.

[Redding 09] Redding, P. 2009. Aarf! Arf arf arf: Talking to the player with barks. *Lecture, Game Developer's Conference 2009*, San Francisco, CA.

[Schwab 14] Schwab, B. 2014. AI postmortem: Hearthstone. *Lecture, Game Developer's Conference AI Summit 2014*, San Francisco, CA.

[Sunshine-Hill 14] Sunshine-Hill, B. 2014. Environmentally conscious AI: Improving spatial analysis and reasoning. *Lecture, Game Developer's Conference AI Summit 2014*, San Francisco, CA.

[Tozour 13] Tozour, P. 2013. From the behavior up: When the AI is the design. *Lecture, Game Developer's Conference AI Summit 2013*, San Francisco, CA.

[Weizenbaum 66] Weizenbaum, J. 1966. ELIZA—A computer program for the study of natural language communication between man and machine. *Communications of the ACM* 9(1): 36–45.

[Wolpaw 07] Wolpaw, E. 2007. Developer commentary. *Portal*. http://theportalwiki.com/wiki/Portal_developer_commentary (accessed January 17, 2015).