Toward Visualization for Games: Theory, Design Space, and Patterns

Brian Bowman, Niklas Elmqvist, *Member, IEEE,* and T.J. Jankun-Kelly, *Member, IEEE*

Abstract—Electronic games are starting to incorporate in-game telemetry that collects data about player, team, and community performance on a massive scale, and as data begins to accumulate, so does the demand for effectively analyzing this data. In this paper, we use examples from both old and new games of different genres to explore the theory and design space of visualization for games. Drawing on these examples, we define a design space for this novel research topic and use it to formulate design patterns for how to best apply visualization technology to games. We then discuss the implications that this new framework will potentially have on the design and development of game and visualization technology in the future.

Index Terms—Computer games, video games, interactive entertainment, entertainment, visualization, game analytics.

1 Introduction

Video and computer games have a long history of using visual representations of data to convey information to the player; for example, colored bars are routinely used to present vitals—health, energy, and magical power—of the player character and are accordingly often called "healthbars" (Figure 1(a)), and many games use overhead maps to show the locations of entities in the game world. Some games have already gone beyond such simple visual representations and are employing statistical data graphics to present more data to the player. For example, the first version of the classic *SimCity* [56] game from 1989 used chloropleth maps to convey metrics such as crime rate in a particular city (Figure 1(b)).

In general, interactive games can be seen as real-time simulations—just like *SimCity*—and there are many situations where feeding back the state of the simulation to the player or the game developer can be beneficial, regardless of game genre. For example, a player of a first-person shooter can use session replays to improve their movement around a map, or a military strategy game player can use statistics about map influence and army strength over time to optimize their use of resources. The ongoing revolution in gaming on social media sites (such as on Facebook) also showcases the power of sharing a player's performance and experience as a social indicator. In addition, data collection and feedback can also be used as a motivational factor; a recent trend in video and computer game design—previously mostly restricted to sports games—is to increasingly instrument games to collect large amounts of data during gameplay and use it to mark special achievements and unlock new content. For example, the multiplayer version of the recent *Call of Duty: Black Ops* [57] first-person shooter collects vast amounts of statistics about player performance, and not only uses this to award special titles, emblems, and equipment to players as they play the game, but also to aggregate all statistics (such as the number of shots fired worldwide, the number of cumulative feet fallen by all players, or the number of vehicles exploded) and feed it back to the entire player community as a social motivator.

We believe that the extraction, management, and visualization of data in games is an exciting new branch of research with huge potential impact—video games sales in the U.S. alone reached $10.5 billion in 2009 [1], and the games industry estimates the number of game consoles in North America to surpass 60 million units (not counting personal computers, which clearly also can be used for gaming) [2]. Furthermore, gamers tend to be highly motivated and often have a high degree of spatial and visual literacy [3]. For these reasons, we think that we are on the cusp of a major adoption of casual visualization [4] technology in millions of living rooms worldwide. The only question is how to best harness visualization technology for the challenges inherent to this exciting new domain.

In this paper, we build on the existing academic research on this topic— [5], [6], [7]—to take the initial steps toward a theoretical framework and methodology for the use of visualization in games. We begin by examining existing evidence of visualization and data collection in games, starting from *SimCity* in 1989 and moving to the *Black Ops* and *Halo: Reach* of present day. We then derive a design space of data and visualization types in the context of games. Based on this design space and on the evidence collected, we propose a set of design patterns for how to best approach games as a visualization researcher or practitioner. Finally, we derive implications that

- *B. Bowman and N. Elmqvist are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN. E-mail: bowmanb@purdue.edu, elm@purdue.edu*
- *T.J. Jankun-Kelly is with Mississippi State University, Starkville, MS. E-mail: tjk@acm.org*

(a) Health (green) and mana bars (blue) for a player character in *World of Warcraft*.

(b) Chloropleth map of crime rate in *SimCity*.

(c) Hit heatmap in *Black Ops*.

(d) Death heatmap in *Halo: Reach* (level Anchor 9).

Fig. 1. Examples of visualizations in genres ranging from role-playing games (*World of Warcraft*), construction and management simulations (*SimCity*), and first-person shooters (*Call of Duty: Black Ops* and *Halo: Reach*).

this work will have on both game and visualization research. These implications show how social motivators often drive the introduction of visualizations in games: player communities for competitive multiplayer games use statistics and visualizations to communicate with each other, share game experiences, and improve their performance. Furthermore, we discuss the benefits of studying visualization for games, both in terms of benefits to games—such as better gameplay, easier balancing and debugging, and more enjoyable spectating—as well as in terms of benefits to visualization research—such as massive adoption, a dedicated user base, and a controlled and easily instrumented data source (the game world).

2 Background

A *computer* or *video game* is an electronic game where the player interacts with a visual display using some form of input device. Stated in this very generic form, it is clear that video games have many commonalities with general interactive systems, such as word processors, web browsers, and spreadsheets. The difference lies primarily in the goal-driven nature of the game, and sometimes in specialized hardware, including dedicated game controllers—such as joysticks, gamepads, steering wheels—or even dedicated video game consoles—such as the current seventh-generation consoles like the Sony Playstation 3, the Microsoft Xbox 360, or the Nintendo Wii.

Although we use the terms *computer game* and *video game* interchangeably, these terms traditionally refer to whether the game is targeted at a personal computer (PC or Mac) or a video game console. This paper is about both of these types of platforms and makes little distinction between them—discrepancies arise only for visualization components outside a game (such as on a website), which generally are targeted for computers, regardless of the actual game platform.

We delimit our treatment in this paper to games, and not real-time interactive simulations for training purposes outside of the entertainment domain. While these share many of the same characteristics, we lack the space to survey this domain as well.

Furthermore, we should emphasize that although virtually all computer games today incorporate advanced and often near-photorealistic graphics, we regard such aspects to fall within the scope of computer graphics (and applied domains such as Virtual Reality) and *not* visualization. In fact, games are one of the driving forces for consumer-level computer graphics, and, by extension, visualization [8], [9]. However, while the 3D rendering of a fictional world within a computer game strictly speaking is a visual representation of data (the 3D world), we delimit this work to focus on the use of visualization to represent data with no intrinsic spatial form (such as health, points over time, political influence, etc). In other words, our treatment here is basically performed from an *information visualization* (InfoVis) viewpoint [10].

2.1 HCI and Games

Games and human-computer interaction (HCI) have a long and colorful history [11]. Already in 1983, Ben Shneiderman used games as illustrative examples in his landmark paper on direct manipulation [12], lauding them for smooth, effortless, and direct interaction. Galloway's classification of action in gaming particularly identify interface elements as key nondiegetic (i.e. outside of the game world) aspects of games [13]. As a result, considerable research has been directed toward applying HCI methods to games [14], [15], [16], [17], but also toward applying techniques from games into HCI [18], [19], [20], [21].

A key difference between general HCI and games is that for the former, the goal is to help users to become experts as quickly and effortlessly as possible, whereas games are generally designed using the motto "easy to learn but difficult to master" [20]. This is an important point; as with HCI, the usability of a visualization should never be a hindrance against players reaching their goals (intrinsic difficulties in the gameplay serve this purpose already).

2.2 Applied Visualization

Visualization for games is still in its infancy, but there is a number of existing visualization technologies that

may be relevant to this novel domain. The target audience of a visualization for a computer game is one of the first issues to consider. If the visualization is targeted primarily at the gamer community, these visualizations will likely fall under the topic of casual information visualization [4]. Such visualizations will need to target attributes such as being "useful" and "pleasing" rather than "utilitarian." If, on the other hand, the target audience is the development team, then techniques from software visualization [22]—such as for log traces, program structure, and runtime behavior—may be more appropriate. Bioware's Skynet [23] is an example of such a developer-oriented telemetry and visualization framework that we will discuss in depth later on in this paper.

Just as for normal visualization, the data governs which technique to use. Because many games tend to be spatial in nature, spatiotemporal visualization techniques are often useful; see Andrienko and Andrienko [24] for an exhaustive survey. Other data may be abstract or purely temporal in nature, and for these we can apply temporal visualization techniques [25] as well as traditional statistical data graphics [26]. Furthermore, multidimensional, graph, and text visualization techniques—currently very uncommon in games—may become more prevalent in the future.

Finally, a closely related, yet equally (if not more) unexplored, research topic is the use of visualization for sports. Sports share many of the characteristics of games in the types of data, the target audience, and the questions that the audience is likely to ask (not to mention the fact that sports games form an extremely popular share of the market). Medler [27] draws connections between the collection of statistics in sports and games; however, visualization is equally underutilized in both domains. TennisViewer [28], which visualizes tennis matches, is one of only a select few examples. More recently, Cox and Stasko [29] use information visualization to discover hidden meaning in baseball season data using bar displays and treemaps. Healey parses play-by-play summaries of NFL football games and combines temporal and spatial representations to visualize them graphically [30].

2.3 Game Telemetry and Analytics

Game telemetry (telemetry is defined as the use of various instruments and sensors to remotely collect real-time measurements) has lately become the catchphrase for all kinds of techniques that collect real-time data on game sessions, and *game analytics* [27], [31] is the practice of analyzing this data. Applications for this analysis are numerous, and include both the developer side (usability, game balancing, profiling, etc) as well as the gamer side (achievements, maps, log analysis, movies, etc). As we shall see in this paper, one approach to game analytics is to use visualization.

Collecting data inside a game requires the developers to *instrument* the source code appropriately, a process that can complicate source code and cause extra overhead. To ease this process, some research and commercial frameworks exist for game telemetry and analytics. For Flash-based games, the key frameworks are Mochibot [32], Nonoba [33], and Playtomic [34] where developers add hooks inside their games to send player events to external servers. Kontagent [35] is targeted at tracking browser-based games running on social media platforms such as Facebook and Myspace. The Microsoft Game Studios TRUE (Tracking Real-Time User Experience) [36] system is used to instrument source code and can then be utilized to automatically collect rich sequences of user events, similar to observational ethnographic studies. The RAD Telemetry Profiling System [37] is targeted at programmers, and collects real-time application performance for the purpose of optimizing game code.

In the absence of standardized frameworks, different games and developers have built custom solutions for game telemetry and analytics. The MMORPG *Pirates of the Burning Sea* [58] uses advanced logging technology to support the design and development process [38]. Similarly, Bioware's role-playing games *Dragon Age: Origins* [59] and *Mass Effect 2* [60] incorporate a developer-oriented telemetry framework called SkyNet [23]. SkyNet collects data from several running instances of a game on the network and presents the data in a web portal. Also, the Maxis game *Spore* [61] provides a gamer-oriented telemetry framework through a public API [39], [40] that allows players to view, analyze, and visualize the data freely and without being constrained by the game itself.

2.4 Visualization and Games

Although not nearly as intimately intertwined with games as HCI, visualization has had its eye on games for some time. However, the primary line of research has been on how to leverage and steer the rapid advances in computer graphics spurred by the video games industry for (mainly scientific) visualization [8], [9]. While this is certainly an important topic for visualization research, our focus in this paper goes beyond computer graphics to study the adoption of actual visualization techniques in video games.

Surprisingly little work has been conducted in this domain. Halper and Masuch [41] focus on extracting and summarizing highlights from real-time gameplay footage to aid spectators in first-person shooters. Hoobler et al. [42] use local visualizations—such as player glyphs, paths, and tracers—in combination with global spatial visualizations to represent team-based competitive gameplay in a multiplayer first-person shooter (*Wolfenstein: Enemy Territory* [62]). These ideas sparked additional work on spatializing movement in 3D environments using visualization [43], [44], [45]. In recent work, Medler et al. [5] present the Data Cracker for the multiplayer version

of the first-person shooter *Dead Space 2* [63] that uses web-based interactive graphs to display information about gameplay behavior.

Unfortunately, the above examples are only point designs for specific games, and it is only recently that visualization research has begun to take a broader view of games as an exciting new domain. Joslin et al. [46] propose a gameplay logging and visualization manifesto, using *World of Warcraft* [64] as an example of how to visualize data from a role-playing game. However, their examples are all interactive mock-ups outside of the actual game. Medler [47] studies how *player dossiers* are starting to be used as containers for many visualization components in games.

Zammitto [7] was among the first to make the connection between visualization and the visual information presented in games that go beyond the 2D or 3D spatial representations of the game world. Her paper reviews and surveys both the academic literature and the video game markets to find evidence for such usage of information visualization. While we build on her initial review in this paper, our work focuses more on the visualization technology aspects of this topic.

In a very recent paper, Medler and Magerko [6] study the use of information visualization in a large number of existing games on the market today. Starting from the concept of casual information visualization [4], they introduce the notion of *playful InfoVis* as the use of information visualization to support and promote play. We have adopted many of the examples and classifications in Medler's and Magerko's work in this paper, but our work instead focuses on the design space, design patterns, and implications for the future design of visualization technology in games.

Finally, another very recent paper, this one by Diakopoulos et al. [48], take the opposite approach compared to ours by incorporating game elements into visualization (referred to as infographics in the paper) instead of vice versa. Their two designs and evaluation show the potential of using playable data to influence and motivate exploration.

3 Evidence of Visualization in Games

The purpose of this section is to review existing games that incorporate some form of visualization components. For the sake of simplicity, we have categorized our review by game genre, informally drawn from online gaming community sites such as IGN (http://www.ign.com/) and GameSpot (http://www.gamespot.com/). In addition to examples, each subsection also includes a description of that genre.

3.1 Game Middlewares and Lobbies

While not strictly games per se, there is a growing trend of game middlewares, gaming communities, and unified gaming lobbies that are beginning to utilize visual representations to convey abstract data in novel ways. For example, the Steam [65] online games distribution and multiplayer gaming platform incorporates rudimentary statistical graphics showing the usage of the platform and various games. For the Xbox 360, playing games will earn the player so-called "achievements", whereas Playstation 3 players earn "trophies", both of which can be shown on the player's profile screen and used in rankings. A recent tutorial [49] by the gaming middleware company GameSpy argued in favor of more advanced visual representations for aggregated game statistics using streamgraphs [50] as an example.

In general, game developers are beginning to extend the reach of the gaming world outside the games themselves onto websites accessible using an online account that is connected to the player's copy of the game. Examples for specific games (all described in detail below) include Bungie.net for *Halo: Reach* [66], FIFA Earth for *FIFA 10* [67], and the online stats for *Call of Duty: World at War* [68] (Figure 2). Because Xbox achievement and PS3 trophy data can be accessed from outside their respective communities, aggregator sites such as GiantBomb.com are starting to combine and visualize this data online, as well as comparing a player's performance with the whole community.

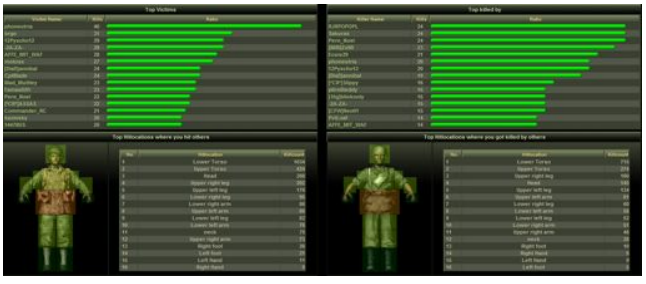

Fig. 2. Example player dossier [47] in *Call of Duty: World at War* [68] (http://www.callofduty.com/stats/).

Community-created websites are also growing as game telemetry data becomes increasingly accessible outside of games. We have already mentioned the god game *Spore* [61] where the developers are providing an actual API for hobby programmers to access in-game data, but for games where such mechanisms are less (or not at all) developed, motivated players are resorting to manual (and tedious) data collection that is then aggregated and visualized on various websites.

3.2 CMS, 4X, and God Games

A popular set of game genres that are particularly amenable to the use of visualization are construction and management simulations (CMS), Explore, Expand, Exploit and Exterminate (4X) games, and god games. Common among all of these is that they put the player in control of a business, empire, or even an entire world from a macro-scale leadership perspective (although micro-scale decisions are often

Fig. 3. City screen in the 4X game *Civilization* [69].

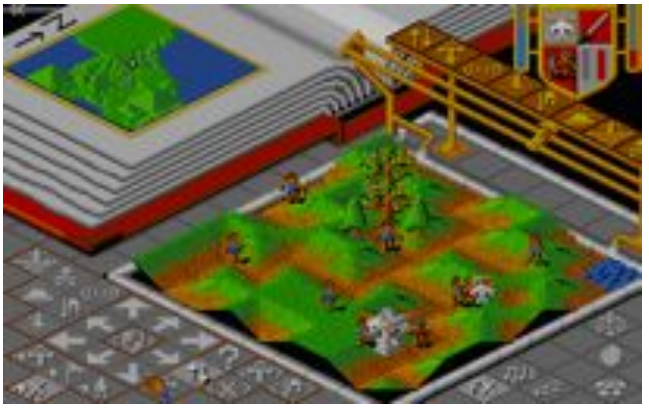

Fig. 4. Game interface in the god game *Populous* [71].

necessary). Because of this high-level perspective, visual representations are easier to integrate into this kind of game while maintaining immersion than into games with more direct and literal gameplay.

Accordingly, some of the first evidence of visualization in games is found in these genres. *SimCity* [56], the canonical CMS game that puts the player in charge of urban planning and administration of an entire city, was released in 1989 and incorporates numerous advanced visual representations such as the choropleth map for spatial data shown in Figure 1(b). Other CMS games, in the Sim series and beyond, have since carried on this tradition by continuing to incorporate visual representations for this type of data.

Another innovator in this area is the *Civilization* series of 4X games designed by Sid Meier, where the player leads an empire spanning through thousands of years and entire worlds. The original *Civilization* [69] came out in 1991, and incorporated novel visual representations such as glyph-based bars (Figure 3) as well as a summary replay of a game session (Figure 5(a)). Later games in the series have continued this tradition, with *Civilization III* [70] in particular introducing several new representations (Figure 5).

God games take the concept a step further by turning the player into a deity with (semi-) absolute control over a world. *Populous* [71] from 1989 was first among such games, and incorporated some rudimentary visual representations, as did *Populous II* [72], its successor (Figure 4). Later offerings, such as the aforementioned *Spore* [61], also often use abstract visual representations for conveying game state.

3.3 First-Person Shooters

First-person shooters (FPS) are action games where the players view and interact with the 3D world through the eyes of the protagonist. With id Software's *Wolfenstein 3D* [73] from 1992 as the genre's spiritual ancestor, FPSes are today one of the most popular game genres on the market, particularly for competitive multiplayer gaming (a subgenre that was arguably sparked by *Doom* [74] in 1993, also by id).

One common feature of most first-person shooters has long been the use of *heads-up displays* (HUDs) (see Section 5.1) that overlay game status such as player health, ammunition, and energy levels on top of the 3D world view. However, as discussed later in this paper, developers are now starting to decrease the size and prominence of HUDs in favor of more immersive mechanisms, such as showing blood spatter, weapon, and ammunition in the game world itself.

Given the focus on immersive first-person gameplay, it is perhaps somewhat surprising that visualization has recently become so prominent in FPS games. However, the explanation may be in the multiplayer and social aspects of these games; multiplayer support is common in FPS games, which often spawns an online community of players that are interested in furthering their own play, communicating with each other, and comparing achievements. Accordingly, there has been much innovation in supporting these communities with visualization: Hoobler et al. [42] (discussed above) visualize data from *Enemy Territory* [62], *Call of Duty: World at War* [68] provides a statistics web portal (Figure 2), *Halo: Reach* [66] sports Bungie.net that tracks and visualizes both aggregated and individual statistics (Figure 1(d)), and *Black Ops* [57] maintains an in-game combat record that shows spatial and temporal statistics (Figure 1(c)).

3.4 Real-Time Strategy

Real-time strategy (RTS) games are a relatively recent type of strategy game, often military- or science fiction-themed, where the player becomes the general of an army and is asked to defeat an enemy. *Dune II* [75] was arguably the first RTS, and introduced the notion of a real-time, continuously updating game world at a time when its strategy brethren were all turn-based. Similar to FPS games, modern RTS games often include an online multiplayer mode (largely influenced by *Warcraft* [76] (1994) and *StarCraft* [77] (1998), the latter which is played professionally).

Perhaps for this reason, statistics and visualization figure heavily in the RTS player communities. *Heroes*

(a) Summary replay. (b) Diplomacy (undirected graph). (c) Technology tree (directed graph).

Fig. 5. Different visualizations for replay, foreign relations, and technological inventions in *Civilization III* [70].

Fig. 6. The "game DNA" of a competitive game in *StarCraft II* [78] from SC2Replayed.com showing the build order of installations for different players on the opposing teams (left vs. right columns).

of Newerth [79] has a web interface that collects player statistics, but little visualization support. Replays are very popular for *StarCraft II* [78], with some important matches even being narrated by professional game commentators. The site SC2Replayed.com collects many replays and also analyzes them to show "DNA" screens of players and games (Figure 6).

Just as for FPS games, HUDs are used prominently in most RTS games. However, because of the fact that RTSes are typically not designed with immersion foremost in mind, RTS games have still not seen the diminishing HUD phenomenon that FPS games have. An alternative explanation is that HUD components such as abstract markers, information readouts, and user interface elements fit better with the bird's eye perspective used by most RTS games than they fit the first-person perspective of shooter games.

3.5 Role-Playing Games

Computer role-playing games (RPGs), like their paper-based counterparts, give the player control of one or several characters that grow in ability as the game progresses. These games are often complex to develop and require careful design to reach the correct balance of challenge and difficulty, making them excellent candidates for visualization in support of developers. Accordingly, the aforementioned *Pirates of the Burning Sea* [58] has a built-in logging and analytics system [38], and Bioware's *Mass Effect 2* [60] (a science fiction RPG) and *Dragon Age: Origins* [59] (a fantasy RPG) were designed using the SkyNet [23] analytics and visualization portal. However, none of these visualization systems are intended for players.

On the player side, many RPGs also sport community-maintained websites with data collections. This is particularly prominent for the subgenre known as MMORPGs, or massively multiplayer online role-playing games, which are large-scale and persistent game worlds for hundreds or thousands of simultaneous players. Again we see the impact of community: MMORPGs such as *World of Warcraft* [64], *Dark Age of Camelot* [80], *Warhammer Online: Age of Reckoning* [81], and *EVE Online* [82] have extensive online repositories, statistics databases, and map collections that have all been painstakingly collected, classified, and chronicled by enthusiasts worldwide.

3.6 Sports Games

Statistics has long been an intrinsic component in professional sports, and so it follows that sports games by extension often also incorporate numerous statistics. However, while sports visualization is growing (see Section 2.2), most games tend to leave the statistics in numerical format and use few in-game visual representations (although in-game tactics and playbooks arguably are a form of visual representation).

Websites outside of the sports games themselves are starting to incorporate visual representations, however. *The New York Times* published a recent infographic [51] showing how the ups and downs of real NFL teams during the season influenced the teams and matchups people were playing in the football game *Madden NFL 11* [83]. FIFA Earth for the *FIFA 10* [67] soccer game is a web-based game portal that aggregates statistics from multiplayer matches to show the macro-scale performance of different countries using visual representations.

3.7 Driving Games

Driving games put the player behind the wheel of different types of vehicles, ranging from sports cars, motorcycles, racing cars, etc. Immersion and realism is often an important aspect of such games, so visualizations are generally rare. However, some visual representations can be integrated even while maintaining

immersion: ghost cars, or *ludophasmas* [27], that show the performance of other players (or the player herself in another session) as an intangible shape on the road have been used ever since the heydays of the old Amiga and Atari computers, and many driving games (starting from the original *Formula One Grand Prix* [84] from 1992) render the optimal racing line on the tarmac to help novices improve their turn-taking.

Social aspects of multiplayer gaming have again had an impact on modern driving games. *Need for Speed: Hot Pursuit* [85] has an in-game component called Autolog that collects racing statistics for each player and visually compares these against the player's friends within the game world.

4 Framework: Visualization in Games

Drawing from the above examples, we now define the design space of visualization for games. Unlike traditional visualization taxonomies [52], we focus only on components that are specific to the intersection between games and visualization technology. For this reason, our design space does not incorporate data type, utility, and visual mappings; such categories are specific to visualization techniques and not their application to games.

Our design framework consists of the following five categories that can be used to classify any specific visualization technique found in a computer game (described in detail in subsequent sections):

- **Primary Purpose:** Intended use;
- **Target Audience:** Intended audience;
- **Temporal Usage:** Timing of use;
- **Visual Complexity:** Level of visual sophistication of the visualization in the game; and
- **Immersion/Integration:** Whether the visualization is in spirit with the game and whether it is inside the game or not.

Note that a single game could contain several different visualization techniques, each having to be classified separately using these dimensions. Furthermore, these visualizations can also be classified using traditional visualization taxonomies.

4.1 Primary Purpose

Visualization techniques are added to games for a reason. The Primary Purpose dimension captures the main purpose that the visualization was created (although some visualizations can arguably have multiple purposes). However, classifying visualizations using this dimension often requires reconstructing the design decisions that went into the game design, which can be difficult and error-prone.

We use the following levels for Primary Purpose:

- *Status:* Status visualizations are designed for conveying important information to the player, typically continuously—examples include healthbars (Figure 1(a)), ammo counters, and hit direction indicators. Visual representations are often chosen in lieu of a simple number (like a percentage for the character's health) because the game designers feel that the visualization is more immersive and easier to read quickly, or even preattentively.
- *Training:* Some visualizations are designed to help players—either the current player, or other players—to improve their gameplay. We classify this as training, and these visualizations often use a reflective mode, such as replay theaters (in *Starcraft II* [78], *Halo: Reach* [66], or *Call of Duty: Black Ops* [57]), but continuous approaches such as a visual overlay showing the optimal line for taking a curve in a driving game are similar.
- *Progression:* Many times players are faced with different choices for how to advance in the game, and designers are increasingly using visualization in these situations to show these options. For example, many games use visual skill, technology or even dialogue trees to show the player how to progress in the game, and the supply and demand bars in *SimCity* [56] tell the user what zones to build in order to grow their city.
- *Communication:* Another purpose for a visualization is for communicating the game or the player's state to others. This practice has become increasingly popular with the rise of social games on sites such as Facebook and Myspace, where status messages posted to one's profile may incorporate visual representations that serve as social indicators of progress. Another recent phenomenon is games as spectating entertainment, where replay theaters and movies are perused by spectators watching professional players compete against each other (for example, this practice is widespread in *StarCraft II* [78], where commentators typically talk over game footage).
- *Debugging and balancing:* Our purposes so far have been mostly focused on the player side of the fence, but as evidenced by our literature survey, developers are also avid consumers of visualizations for game analytics. Such visualizations are commonly designed for allowing the developer to debug, balance, and compare player performance between and within different sessions; they are therefore often not integrated within the game, they aggregate several game sessions, and they may also rely on a higher visual literacy on behalf of the users. Examples include Flying Lab Software's logging technology for *Pirates of the Burning Sea* [58] or Bioware's Skynet [23] portal that aggregates large amounts of gameplay data to enable in-depth analysis.

4.2 Target Audience

We use Target Audience to model the intended users of a visualization in a computer game. Our division

into levels here is very rough; Medler and Magerko [6] use Brown's classification of player types [3] to discuss the appropriateness of analytical interactions for different audiences. We recommend their approach for delving into player personalities. Our framework has the virtue of being simpler and also including non-gamers, one goal of this work:

- *Player:* Most visualizations are designed for players, and may either be of a precise nature (such as skill and tech trees), or instances of casual information visualization [4] (or even what Medler and Magerko [6] calls "playful information visualization"). The purpose of this visualization type is not necessarily optimal performance (although it may be for competitive multiplayer games), and conforming to a player often also means that the visualization is in spirit with the game's atmosphere and integrated within the game.
- *Developer:* Visualization has lately become a popular tool among developers for game analytics of telemetry data for debugging, balancing, and playtesting the game. An early example is the logging and analytics system used in *Pirates of the Burning Sea* [58]; a particularly impressive one is Bioware's SkyNet [23] tool that integrates a web-based analytics portal with the game.
- *Observer:* A growing trend in the gaming community is not only playing games, but also watching other people play them. Competitive multiplayer gaming is now a spectator sport, with first-person shooters such as *Quake* [86] and real-time strategy games such as *StarCraft* [77] leading the way, and the needs of an observer are very different from those of a player. Visualization can help; for example, Halper and Masuch [41] use visual techniques to automatically create a highlight reel from an FPS session.

4.3 Temporal Usage

The first visual depictions of game state were probably score and healthbars, and these were designed to always be visible in order to give the player continuous feedback about their character's state. However, as visualization in games matures, we are starting to see other timing aspects for its use. We capture these aspects in the Temporal Usage dimension, and we define it as having the following values:

- *Continuous:* The most common level is continuous representation, where the visualization is (more or less) constantly updating and provides live feedback about the game state. The canonical example of a continuous visual representation is the ubiquitous healthbar and its variations (ammobar, powerbar, energybar, etc); see Figure 1(a). Several such continuous representations are often integrated into *heads-up displays* [7].
- *Intermittent:* Some visualizations are designed to not always be visible and instead for being called up intermittently during gameplay, often because they are large and potentially distracting. Examples of such representations include in-game maps for first-person games, inventory screens that show the character's current equipment, or heads-down views in flight simulators.
- *Retrospective:* The newer breed of visual representations in games are designed for retrospective use, often outside of the game or the main gameplay, and allow for the user to study their performance in a post-mortem fashion. The purpose of this kind of visualization is often to help the player improve their skills, for example using replay theaters in games such as *Call of Duty: Black Ops* [57] and *Halo: Reach* [66], but other uses are just for revealing all of the game state to the user after the game, like in summary replays for the *Civilization* [69] series of games.
- *Prospective:* Finally, some visual representations are designed to help the user make predictions about the future. A simple example is the dialog system in *Dragon Age II* [87] that uses small glyphs next to each dialog option to indicate its emotional tone, or the practice of color-coding the names of monsters (i.e., what is shown when the user selects a monster to potentially attack) in many role-playing games to indicate the risk of failure if the user would try to kill the monster.

4.4 Visual Complexity

The Visual Complexity captures the level of visual sophistication of the visualization that is embedded in the game. It can be argued that this is a feature of the visualization itself and not its application to computer and video games, but we think it is important because it gauges the level of visual literacy that the game designers are attributing to the players. Another issue that may be interesting to study is whether the level of visual complexity is increasing with time, with newer games including more advanced visual representations than older ones.

- *Basic:* Very simple visual representations, with healthbars, powerbars, and ammo counters being the canonical example. This level of visual complexity has basically existed within games since the inception of digital games.
- *Intermediate:* This level represents visual representations that use more advanced mappings that go beyond the standard visual variables of length, size, color, position, etc [53]. We count most standard statistical data graphics (such as the score and kill statistics in *Call of Duty: Black Ops* [57]), illustrative animations (such as the summary replay in *Civilization* [69]), and in-game sprites

for guidance or recorded gameplay (known as *ludophasmas* [27]) as belonging to this category.

- *Advanced:* Finally, we also survey techniques that use an advanced visual language, thus assuming a high degree of visual literacy on behalf of the users. While the choice between intermediate and advanced complexity sometimes can be debatable, we would classify advanced representations to be those that casual users would not often come across in their daily lives. Given this guideline, we classify the chloropleth maps in *SimCity* [56], diplomacy graphs showing relations between factions (commonly used in the *Civilization* [69] series and other 4X games), and basic hierarchical representations for character progression (such as skill trees in *Diablo II* [88] or talent trees in *World of Warcraft* [64]) as part of this class.

4.5 Immersion/Integration

Finally, we believe it to be useful to capture the visualization's immersion factor in the game, as well as whether or not it is part of the game application itself or is found outside the game proper. These are clearly not orthogonal factors—a visualization separate from the game itself is naturally less immersive than one inside the game—but we think it is useful to keep this distinction to capture the full spectrum of visualization in games and their application.

- *Immersive/Integrated:* These visualizations are implemented inside the game itself, and they are in spirit with the game's atmosphere. The latter part naturally depends on the game genre—for example, line charts and pie diagrams are fully appropriate for a business simulation game, but not necessarily so for a first-person shooter—and must therefore be judged on the visualization's application inside the game rather than on the visualization itself. One example of an interesting immersive/integrated visual representation is the hit direction indicator found in almost all modern first-person shooters; it shows the rough direction the player is taking fire from, but we still regarded it as immersive because it mimics information that the character would have but which is difficult to convey to the player.
- *Informative/Integrated:* Other visual representations are added to a game for the purpose of providing additional information to the player. They are still implemented inside the game application, but they often reveal information about game state to the player that the player's character would have no access to inside the game world. Examples include replay theaters and performance charts over time in an FPS. It should be noted that many cheats fall inside this category; for example, so-called "wallhacks" in FPS remove walls in levels so that players can see opponents through them, and "radar" tools reveal enemy positions on a map (note that some games use radars as immersive components of gameplay, so classification depends on the game).
- *Immersive/Separate:* Although being separate from the game itself would seem to suggest that these visual representations cannot be immersive, there still exists such examples in the market. One such is *World of Warcraft Remote* [64] (http://us.battle.net/wow/en/services/wow-remote/), which allows players to chat in-game and participate in in-game item auctions using their smartphone.
- *Informative/Separate:* Finally, we also include visual representations that are implemented outside of the game. This case has received little discussion so far in this paper, but represents a significant trend on the Internet as game developers open up their data APIs so that hobby gamers can analyze (and visualize) them. Examples include annotated monster maps for games such as *World of Warcraft* [64] and *Dark Age of Camelot* [80], the data API [40] for *Spore* [61], and the player-drawn political maps in *EVE Online* [82].

5 Design Patterns

In this section we show how to use our new framework by aggregating common uses of visual representations in games into *design patterns* [54] and then classifying them using the taxonomy—see Table 1. We focus on five high-level categories of patterns in this text, each grouping together several actual patterns.

5.1 Heads-Up Display

A *heads-up display* [7] (HUD) is a common term for all in-game visual representations that are integrated with the live gameplay; in particular, the HUD often consists of healthbars, magic bars, and ammunition counters as well as interactive maps, toolbars, and other interface components. All of these visual representations are overlaid on top of the 2D or 3D view of the game world using transparency.

Despite their widespread use, it may be argued that HUDs are not optimal designs because they exist outside of diegetic space (i.e., the space of the story). For this reason, an increasing number of games are reducing the size of their HUDs, or removing them altogether; for example, RPGs such as *Dragon Age II* [87] and *Mass Effect 2* [60] only show healthbars when in combat, and only then when characters have taken damage, and many first-person shooters such as *Call of Duty: Modern Warfare 2* [89] and *Battlefield: Bad Company 2* [90] are replacing healthbars with making the player's screen flash red, darken at the edges, and grow blurry as the player takes damage (Figure 7).

Some of the common low-level design patterns for heads-up displays include the following:

Pattern Group	Primary Purpose	Target Audience	Temporal Usage	Visual Complexity	Immersion/ Integration	Examples
Heads-Up Display	status	player	continuous	basic	imm/int	common
Replay Theater	training/comm	player/observer	retrospective	basic	inf/–	[57], [78]
Progression Tree	progression	player	prospective	advanced	inf/int	[69], [70], [88]
Data in Space	train/comm/status	player/observer	retrospective	inter/advanced	inf/–	[56], [57], [66]
Data in Time	train/comm/status	player/observer	retrospective	inter/advanced	inf/–	[66], [69], [70]

TABLE 1
Common design pattern groups for visualization in games. Fields marked as '–' change between designs.

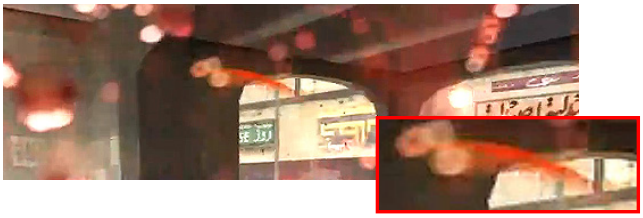

Fig. 7. Immersive heads-up display in *Call of Duty: Modern Warfare 2* [89] where the blood on the screen indicates that the player has taken damage, and the red fan shape (inset) is the hit direction indicator.

- **Information readout:** Basic visual representations of game state, such as score, healthbars, powerbars, compass heading, current target, etc.
- **Crosshairs:** Common in first-person shooters, crosshairs augment the visual display with an indication of where exactly the player character's weapon is aimed. This is different from using the actual weapon sights for this purpose, in which case the crosshairs are part of the game world and not a HUD component.
- **Hit indicator:** A direction indicator that is shown on the screen whenever the player is being hit by gunfire to indicate roughly from which direction the enemy fire is coming (Figure 7).

There also exist immersive (i.e., that are integrated in the game world) HUD design patterns:

- **Cockpit:** Some genres (particularly simulations) lend themselves to using the actual cockpit or dashboard of the in-game vehicle to communicate information, for example using cockpit gauges and meters to show speed, heading, and fuel.
- **World annotation:** These add visual representations to the world itself, such as the optimal racing line in driving games, and directional arrows and highlighted objects in 3D environments.
- **Ludophasmas:** In-game sprites that guide the player or communicate previously recorded gameplay inside the world, but which still are narrative elements (see Medler [27] for details).

5.2 Replay Theater

Replay theaters are mechanisms for post-mortem analysis of a game session by playing back the session, either for players to see their own performance or for observers that are spectating on a match. Developers may also want to use replays to catch bugs and game design issues (it is integrated into the SkyNet [23] system). Gaming, particularly multiplayer gaming, has a long history of recording game sessions as videos (a practice more generally known as *machinima* [55]), but these restrict the viewer to just watching a non-interactive animation. Replay theaters take this a step further by recording telemetry for all entities in a game session, not just the screen of one player.

- **Time controls:** The addition of time controls relax the temporal constraints of the live gameplay and allows the viewer to navigate the recorded session in time, as well as freeze time at will.
- **Space controls:** Like time controls, the space controls pattern for a replay theater allows the user to freely navigate in the game environment to study the gameplay action from different viewpoints.
- **Event timeline:** Timeline visualizations can be integrated with the replay theater to aid navigation in the replay data; for example, the theater mode in the recent *Call of Duty: Black Ops* [57] includes a timeline visualization that shows player deaths and kills over the course of the session (Figure 8).

Fig. 8. Playback control interface for the replay theater in *Call of Duty: Black Ops* [57]. Note the timeline on top that visualizes the player's kills and deaths as green and red lines at different positions in time.

5.3 Progression Tree

Progression trees are directed graphs (or trees, in the special case) modeling dependencies between concepts to allow for player progression in a game. These graphs are often visualized in games to make these dependencies visible to players and allow them to plan ahead. Since the graphs are generally fixed and do not change dynamically, graph layout algorithms are typically not needed in the actual game.

- **Skill tree:** Many games that are focused on character progression, such as RPGs, use *skill*

trees to let the player customize their character's abilities. The skill tree thus captures the prerequisites needed to learn a particular skill. Examples include the skill system in Blizzard's *Diablo II* [88] and the talent system in *World of Warcraft* [64],

- **Technology tree:** This pattern is used to capture progression of technology in 4X games; for example, in the game *Civilization* [69], the technologies Horseback Riding and Feudalism are both needed in order to research Chivalry. Figure 5(c) shows part of the tech tree in *Civilization III* [70].

5.4 Data in Space

Patterns in the *data in space* group are visual representations of spatial game data. This is not a new phenomenon in games, but it is increasingly gaining traction outside of the CMS and 4X genres.

- **Annotated map:** Mapmaking of game worlds [6] has always been common for RPGs, and this practice is even more widespread in communities for MMORPGs such as *EVE Online* [82], *Dark Age of Camelot* [80], and *World of Warcraft* [64].
- **Influence map:** This pattern creates color-coded political maps of space to show the influence and territory of different factions in a game. For example, the *Civilization* [69] franchise includes so-called "histographs" (Figure 5(a)) that show the respective influences of different empires over continents as a space-filling visualization.
- **Heatmap:** These thematic maps convey quantity of a statistical variable over spatial units, such as the chloropleth maps in *SimCity* [56]. Perhaps the most exciting development here are the spatial visualizations used in FPS games, including Bungie's *Halo: Reach* [66] and Treyarch's *Black Ops* [57], that aggregate millions of multiplayer games to visualize where players tend to die and to kill other players in different game levels. Figure 10 shows death and kill heatmaps for seven levels (provided by Bungie.net), and then show the difference map that we have constructed which indicates areas on each map where players tend to be at an advantage (i.e., more players kill other players here than are killed).
- **Fog of War:** Modern strategy games often employ the concept of *fog of war* where the player is only made aware of enemy movement in areas to where one of their units has direct line-of-sight, and the visual representations of this concept is an instance of a data in space pattern. In practice, this is often implemented by only showing actively viewed areas of a game world in full detail, and blacking out the unexplored areas and greying out the explored but currently not actively viewed areas (this is the case in the real-time strategy game *Warcraft* [76]).

5.5 Data in Time

This group of design patterns convey information about game state as a function of the temporal dimension, and for this reason these patterns are often retrospective and informative in nature:

- **Event timeline:** We discussed event timelines as part of the replay theater, but these have a more general use for conveying discrete event sequencies over time. For example, SC2Replayed.com renders build order over time (Figure 6) for *StarCraft II* [78] multiplayer matches.
- **Time-series chart:** This pattern models the visual representation of one or several statistical variables over the temporal domain. Business simulation games regularly incorporate sales diagrams and other forms of data in time, and exercise games such as *Wii Fit* [91] often visualize player performance as a function of time. *Call of Duty: Black Ops* [57] has a player dossier [47] system containing several time-series charts (Figure 9).

Fig. 9. Wager earnings over time in the Combat Record for competitive multiplayer performance of a player in *Call of Duty: Black Ops* [57].

5.6 Additional Patterns

Our above list of design patterns is far from exhaustive and merely represents the most common such patterns in a first attempt to standardize a simple pattern catalogue. Several additional design patterns exist: for *player dossiers* [47], which aggregate metrics about a player's performance; for *mission planners*, which allow players (and/or game designers) to plan scenarios, e.g., for a military simulation involving many units; or for *status reports*, which show the current status of the game as a series of reports, such as the economy, technology, and diplomacy advisor screens (Figure 5) in the *Civilization* [69] franchise.

6 Implications

There are many implications to be drawn from our work, both for improving games using visualization as well as for improving visualization using games. Below we review some of these implications and benefits. However, we start with the effect of community on visualization for games.

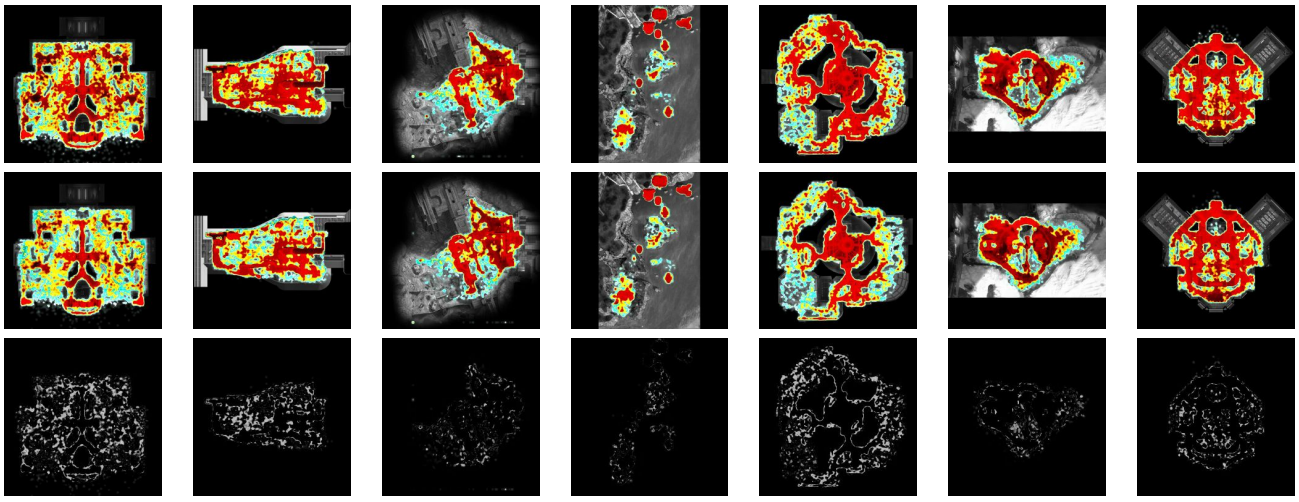

Fig. 10. Global heatmaps for deaths (top row), kills (middle row), and their difference (bottom row) for seven multiplayer maps in *Halo: Reach* [66]. The difference map (our contribution) represents areas in the map where, statistically speaking, you are at an advantage to kill other players but not be killed yourself.

6.1 Social Motivators

In surveying the repertoire of games with visualization components, one trend is obvious: games supporting multiplayer are disproportionally represented in this list. The reason for this is that multiplayer games create a community of players interacting with each other inside the game, and this drives a need for communicating with each other, sharing game experiences, and comparing oneself to other players.

Having a strong community also causes a player's investment of time and effort in a game to increase, thereby giving an incentive to players to create their own visualization components using live telemetry data provided by the game developers. The benefits are clearly major, and we hope that more game developers in the future follow the example of games such as *Spore* [61] and *Halo: Reach* [66] in providing public APIs for accessing game statistics and telemetry.

6.2 Potential Benefits to Games

A large portion of this paper has been about showcasing the benefit of applying visualization in games:

- **Players** can use visualization to improve their skills, be more aware of the game world and its state, and communicate their achievements and progress to fellow players;
- **Developers** can use visualization to detect bugs, find game balance issues, and draw inspiration from player behavior;
- **Observers** can use visualization to draw more enjoyment from watching others play through better access to game data.

We look forward to seeing how future work in this design space can extend and improve games in directions not covered in this paper.

6.3 Potential Benefits to Visualization

One obvious question in all this is what the benefit of applying visualization to games is to us as visualization researchers. While it can be argued that games have a long way to go until they start incorporating advanced visualization techniques, let alone begin to influence research in visualization as a field, the benefit of this approach lies more in adoption and impact than scientific progress. With more than 60 million video game consoles in North America alone, the video and computer game industry represents a huge base of potential casual users scattered in living rooms worldwide. As mentioned early on in this paper, we believe that we are on the cusp of a major adoption of visualization on a large scale, and visualization researchers should be part of this trend to help shape and guide it in the right direction.

Gamers are an interesting population because they are often very passionate and willing to spend time on improving their own gameplay, or sometimes just watch other people play. In addition, they often have a high degree of visual literacy and spatial ability [3]. Integrating visualization components in games represents a unique opportunity for tapping into this population and exposing it to our technologies.

As stated earlier in this paper, one potential synergy for games is the equally unexploited sports domain, which also shares many of the same challenges, audience, and opportunities—not to mention data types (primarily spatiotemporal data and simple statistics). One benefit may thus be that advances in one domain could also be applied to the other domain.

Finally, while visual representations in games currently admittedly are simplistic, this is not to say that researchers cannot make significant progress by conducting visualization research related to games; in

fact, many of the visualization problems encountered in games are reflections of similar problems encountered in the real world. The difference may be that the game world can be more easily controlled and instrumented than the real world. For this reason, we certainly believe there is ample opportunity for deriving new visualization techniques for game data that may become more broadly applicable to real-world data and real-world problems as well.

6.4 Limitations and Improvements

The framework proposed in this paper is consistent with earlier work by Zammitto [7] as well as Medler and Magerko [6], but extends these works in significant new ways with categorical dimensions, practical design patterns, and future implications. Furthermore, we have successfully used the framework to classify and categorize five design patterns that are commonly recurring in game design. While no design space framework is ever truly complete, these are still good indications that our framework has sufficient expressive power for capturing these design patterns.

Having said that, there is certainly room for improvements and refinements in the framework. For example, the only real concession to the nature of visual representation is the "Visual Complexity" category; many more visual features could be modeled here. On the other hand, there is a value to keeping the framework orthogonal to general visualization taxonomies [52] to minimize complexity, which future improvements should try to maintain.

7 Conclusions and Future Work

Drawing on a wide variety of examples from the last two decades of video game history, we formulate the design space of visualization for games as an exciting new branch of casual information visualization with the potential for broad impact in millions of living rooms worldwide. We proceed to use our framework to identify common design patterns for how to employ visualization in computer and video games.

It is clear that both the visualization and the game development communities so far only have scratched the surface of this exciting new problem domain. Our future work will continue mapping out interesting examples that will help progress in both fields.

References

[1] S. E. Siwek, "Video games in the 21st Century: The 2010 report," Entertainment Software Association, Tech. Rep., 2010.

[2] Ipsos MediaCT, "Essential facts about the computer and video game industry: 2010 sales, demographic and usage data," Entertainment Software Association, Tech. Rep., 2010.

[3] S. Brown and C. Vaughan, *Play: How It Shapes the Brain, Opens the Imagination, and Invigorates the Soul*. Avery, 2009.

[4] Z. Pousman, J. T. Stasko, and M. Mateas, "Casual information visualization: Depictions of data in everyday life," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1145–1152, 2007.

[5] B. Medler, M. John, and J. Lane, "Data Cracker: Developing a visual game analytic tool for analyzing online gameplay," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 2011, pp. 2365–2374.

[6] B. Medler and B. Magerko, "Analytics of play: Using information visualization and gameplay practices for visualizing video game data," *Parsons Journal for Information Mapping*, vol. III, no. 1, pp. 1–12, 2011.

[7] V. Zammitto, "Visualization techniques in video games," in *Proceedings of Electronic Information, the Visual Arts, and Beyond*, 2008, pp. 267–276.

[8] T.-M. Rhyne, "Computer games' influence on scientific and information visualization," *IEEE Computer*, vol. 33, no. 12, pp. 154–156, 2000.

[9] ——, "Computer games and scientific visualization," *Communications of the ACM*, vol. 45, no. 7, pp. 40–44, Jul. 2002.

[10] S. K. Card, J. D. Mackinlay, and B. Shneiderman, Eds., *Readings in Information Visualization: Using Vision to Think*. San Francisco: Morgan Kaufmann Publishers, 1999.

[11] Truna aka J. Turner and D. Browning, "Workshop on HCI and game interfaces: A long romance," in *Proceedings of the Austalian HCI Conference*, 2010.

[12] B. Shneiderman, "Direct manipulation: A step beyond programming languages," *Computer*, vol. 16, no. 8, pp. 57–69, 1983.

[13] A. R. Galloway, *Gaming: Essays On Algorithmic Culture*. Minneapolis: University of Minnesota Press, 2006.

[14] P. Barr, J. Noble, and R. Biddle, "Video game values: Human-computer interaction and games," *Interacting with Computers*, vol. 19, no. 2, pp. 180–195, 2007.

[15] R. Bernhaupt, *Evaluating User Experience in Games: Concepts and Methods*. London: Springer, 2010.

[16] D. Pinelle, N. Wong, and T. Stach, "Heuristic evaluation for games: Usability principles for video game design," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 2008, pp. 1453–1462.

[17] P. Zaphiris and C. S. Ang, "HCI issues in computer games," *Interacting with Computers*, vol. 19, no. 2, pp. 135–139, 2007.

[18] J. Dyck, D. Pinelle, B. Brown, and C. Gutwin, "Learning from games: Hci design innovations in entertainment software," in *Proceedings of Graphics Interface*, 2003, pp. 237–246.

[19] C. Johnson, "Using cognitive models to transfer the strengths of computer games into human computer interfaces," in *Proceedings of the Workshop on Fun and Human-Computer Interaction*, 1998.

[20] T. W. Malone, "Heuristics for designing enjoyable user interface: Lessons from computer games," in *Proceedings of the ACM Conference on Human Factors in Computer Systems*, 1982, pp. 63–68.

[21] R. Pausch, R. Gold, T. Skelly, and D. Thiel, "What HCI designers can learn from video game designers," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 1994, pp. 177–178.

[22] J. T. Stasko, J. Domingue, M. H. Brown, and B. A. Price, Eds., *Software Visualization — Programming as a Multimedia Experience*. The MIT Press, 1998.

[23] G. Zoeller, "Development telemetry in video games projects," in *Proceedings of the Game Developers Conference*, 2010.

[24] N. Andrienko and G. Andrienko, *Exploratory Analysis of Spatial and Temporal Data: A Systematic Approach*. Springer-Verlag, 2006.

[25] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," in *Proceedings of the IEEE Symposium on Visual Languages*, 1996, pp. 336–343.

[26] W. S. Cleveland, *Visualizing Data*. Summit, NJ, USA: Hobart Press, 1993.

[27] B. Medler, "Generations of game analytics, achievements and high scores," *Journal for Computer Game Culture*, vol. 3, no. 2, pp. 177–194, 2009.

[28] L. Jin and D. C. Banks, "TennisViewer: A browser for competition trees," *IEEE Computer Graphics & Applications*, vol. 17, no. 4, Jul.–Aug. 1997.

[29] A. Cox and J. Stasko, "SportVis: Discovering meaning in sports statistics through information visualization," in *Conference Compendium of the IEEE Symposium on Information Visualization*, 2006, pp. 115–116.

[30] C. G. Healey, "Visualizing NFL football games," http://www.csc.ncsu.edu/faculty/healey/NFL_viz/, 2008.
[31] A. Drachen and A. Canossa, "Towards gameplay analysis via gameplay metrics," in *Proceedings of the International MindTrek Conference*, 2009, pp. 202–209.
[32] "Mochibot," http://www.mochibot.com/.
[33] "Nonoba: Statistics & tracking," http://nonoba.com/developers/statistics.
[34] "Playtomic," http://playtomic.com/.
[35] "Kontagent: Social analytics," http://www.kontagent.com/.
[36] J. H. Kim, D. V. Gunn, E. Schuh, B. Phillips, R. J. Pagulayan, and D. R. Wixon, "Tracking real-time user experience (TRUE): a comprehensive instrumentation solution for complex systems," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 2008, pp. 443–452.
[37] RAD Game Tools, "Telemetry profiling system," 2010, http://www.radgametools.com/telemetry.htm.
[38] J. Ludwig, "Flogging: Data collection on the high seas," in *Proceedings of the Game Developers Conference*, 2007.
[39] D. Moskowitz, S. Shodhan, and M. Twardos, "Spore API: accessing a unique database of player creativity," in *SIGGRAPH Talks*, 2009.
[40] M. Twardos, "Spore data," http://thevioletpiece.com/spore/.
[41] N. Halper and M. Masuch, "Action summary for computer games: Extracting action for spectator modes and summaries," in *Proceedings of the International Conference on Application and Development of Computer Games*, 2003, pp. 124–132.
[42] N. Hoobler, G. Humphreys, and M. Agrawala, "Visualizing competitive behaviors in multi-user virtual environments," in *Proceedings of the IEEE Conference on Visualization*, 2004, pp. 163–170.
[43] L. Chittaro, R. Ranon, and L. Ieronutti, "VU-Flow: A visualization tool for analyzing navigation in virtual environments," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1475–1485, Nov./Dec. 2006.
[44] R. Thawonmas and K. Iizuka, "Visualization of online-game players based on their action behaviors," *International Journal of Computer Games Technology*, 2008.
[45] R. Thawonmas, M. Kurashige, and K.-T. Chen, "Detection of landmarks for clustering of online-game players," *International Journal of Virtual Reality*, vol. 6, no. 3, pp. 11–16, 2007.
[46] S. S. Joslin, R. A. Brown, and P. Drennan, "The gameplay visualization manifesto: a framework for logging and visualization of online gameplay data," *Computers in Entertainment*, vol. 5, 2007.
[47] B. Medler, "Player dossiers: Analyzing gameplay data as a reward," *International Journal of Computer Game Research*, vol. 11, no. 1, Feb. 2011.
[48] N. Diakopoulos, F. Kivran-Swaine, and M. Naaman, "Playable data: Characterizing the design space of game-y infographics," in *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 2011.
[49] T. Northcutt, "Leaderboards can suck it! five better ideas for visualizing game data," in *Proceedings of the Game Developers Conference*, 2010.
[50] L. Byron and M. Wattenberg, "Stacked graphs - geometry & aesthetics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1245–1252, 2008.
[51] K. Quealy, "Gamers mimic the season's ups and downs," *The New York Times*, Feb. 2011.
[52] S. K. Card and J. Mackinlay, "The structure of the information visualization design space," in *Proceedings of the IEEE Symposium on Information Visualization*, 1997, pp. 92–99.
[53] J. Bertin, *Sémiologie graphique: Les diagrammes - Les réseaux - Les cartes*, Paris, France, 1967.
[54] C. Alexander, S. Ishakawa, and M. Silverstein, *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press, 1977.
[55] Machinima.com, http://www.machinima.com/.

GAME EXAMPLES

[56] Maxis Software, *SimCity*. Brøderbund, 1989.
[57] Treyarch, *Call of Duty: Black Ops*. Activision, 2010.
[58] Flying Lab Software, *Pirates of the Burning Sea*. SOE, 2008.
[59] Bioware, *Dragon Age: Origins*. Electronic Arts, 2009.
[60] ——, *Mass Effect 2*. Electronic Arts, 2010.
[61] Maxis, *Spore*. Electronic Arts, 2008.
[62] Splash Damage, *Wolfenstein: Enemy Territory*. Activision, 2003.
[63] Visceral Games, *Dead Space 2*. Electronic Arts, 2011.
[64] Blizzard, *World of Warcraft*. Blizzard, 2004.
[65] Valve Corporation, *Steam*. Valve Corporation, 2003.
[66] Bungie, Inc., *Halo: Reach*. Microsoft Game Studios, 2010.
[67] EA Canada, *FIFA 10*. EA Sports, 2009.
[68] Treyarch, *Call of Duty: World at War*. Activision, 2008.
[69] MicroProse, *Civilization*. MicroProse, 1991.
[70] Firaxis Games, *Civilization III*. Infogrames, 2001.
[71] Bullfrog, *Populous*. Electronic Arts, 1989.
[72] ——, *Populous II: Trials of the Olympian Gods*. EA, 1991.
[73] id Software, *Wolfenstein 3D*. Apogee Software, 1992.
[74] ——, *Doom*. id Software, 1993.
[75] Westwood Studios, *Dune II: Battle for Arrakis*. Virgin, 1992.
[76] Blizzard, *Warcraft: Orcs & Humans*. Blizzard, 1994.
[77] ——, *StarCraft*. Blizzard, 1998.
[78] ——, *StarCraft II: Wings of Liberty*. Blizzard, 2010.
[79] S2 Games, *Heroes of Newerth*. S2 Games, 2010.
[80] Mythic, *Dark Age of Camelot*. Mythic Entertainment, 2001.
[81] ——, *Warhammer Online: Age of Reckoning*. EA Games, 2008.
[82] CCP Games, *EVE Online*. CCP Games, 2003.
[83] EA Tiburon, *Madden NFL 11*. EA Sports, 2010.
[84] MicroProse, *Formula One Grand Prix*. MicroProse, 1992.
[85] Criterion Games, *Need for Speed: Hot Pursuit*. EA Games, 2010.
[86] id Software, *Quake*. GT Interactive, 1996.
[87] Bioware, *Dragon Age II*. Electronic Arts, 2011.
[88] Blizzard North, *Diablo II*. Blizzard, 2001.
[89] Infinity Ward, *Call of Duty: Modern Warfare 2*. Activision, 2009.
[90] DICE, *Battlefield: Bad Company 2*. Electronic Arts, 2010.
[91] Nintendo, *Wii Fit*. Nintendo, 2007.

Brian Bowman is a senior undergraduate student in Computer Engineering at Purdue University in West Lafayette, IN, USA. He will be graduating from Purdue in Spring 2012. He is a member of IEEE-HKN. He probably plays too many video games.

Niklas Elmqvist is an assistant professor in the School of Electrical and Computer Engineering at Purdue University in West Lafayette, IN, USA. He was previously a postdoctoral researcher at INRIA in Paris, France. He received his Ph.D. in 2006 from Chalmers University of Technology in Göteborg, Sweden. He is a member of the IEEE and the Computer Society. In another life, he is a wizard, assassin, or knight in whatever game currently takes his fancy.

T.J. Jankun-Kelly is an associate Professor of computer science and engineering within the James Worth Bagley College of Engineering, Mississippi State University, MS, USA. His research lies at at the intersection of scientific and information visualization. He received his PhD from the University of California, Davis, in 2003 and is a member of IEEE and the Computer Society. He, too, probably plays too many video games.