Gameplay Driven Content Generation

John P. Benge

Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Fine Arts in Interactive Design and Game Development

At

Savannah College of Art and Design

John P. Benge

Author	Sign	Date

Aram Cookson

Committee Chair	Sign	Date

Josephine Leong

Committee	Sign	Date

Gregory Johnson

Committee	Sign	Date

Gameplay Driven Content Generation

A Thesis Submitted to the Faculty of the Department of Interactive Design and Game Development in Partial Fulfillment of the Requirements for the Degree of Master of Fine Arts in Interactive Design and Game Development

at

Savannah College of Art and Design

By

John P. Benge

Savannah, GA

November 2013

Table of Contents

Index of Figures

Gameplay Driven Content Generation

John P. Benge

November 2013

An increasing number of games are embracing the concept of user generated content as a means of extending their lasting appeal and strengthening their potential for success. Of the millions of people who play these games however, a relatively small percentage actively participate in content creation. A large part of this disparity is a result of the fact that content creation is not, in its current form, as fun or engaging to the player as playing the game itself. The process of generating content for games has been, until this point, a wholly separate process from playing games.

With the aid of procedural generation, it is possible to create a system wherein players act as content creators while simultaneously playing the game. The very act of playing the game in its typical manner results in the creation of content for other players to experience. Through research and implementation, I intend to prove that such an experience can be created. This system of generating levels as a direct consequence of gameplay creates an ecosystem wherein all players become active content creators.

Thesis Statement

Prolonged replayability can be achieved by creating a system in which environments are generated in response to player actions, thus enabling new levels to be created each time the game is played.

1. Introductions

Video games are increasingly becoming defined as experiences rather than self-contained products. To create a game and release it into the marketplace is no longer enough to ensure it becomes successful. An increasingly important component of making a successful game is making the experience last as long as possible; extending the experience so that the player gets enjoyment from the product over a longer period of time. Developers are finding new ways to extend the experience beyond the base product using strategies such as offering downloadable content to extend or expand the game experience or additional services that serve as supplements to the game itself. The goal of these efforts is to keep players engaged with a game as long as possible, keeping them playing and talking about a game for years after it is released. Another strategy for holding players' attention that has been in use for many years, yet still offers untapped potential to extend the game experience is allowing users to develop and distribute their own content for the game.

User generated content is a concept that allows players to create their own content and integrate it within the game world. Allowing the community to create its own content often involves giving players the same tools, or at least tools similar to those used by developers to create the game. Players can then use these tools to create content and add it to their own versions of the game. User generated content can take various forms and can range from the creation of new characters or items, to the creation of new levels, or even the implementation of altered or entirely new game mechanics. This idea is not new. Since the very first computer games, users have been fascinated with the idea of altering the way a game looks or behaves. Players would sometimes tamper with a game's code and alter it, even without the consent of the game's developers. As games evolved, some game developers embraced the idea of allowing users to change their games. This led to many games being shipped with tools allowing users to modify the game and change their experience. Providing players with the means to change or otherwise alter their game experience has garnered a very positive reaction from players, and many developers now understand its potential for adding to the value of their product. It is now fairly common for a game to not only allow for user generated content, but to highlight the feature as one of its marquee selling points.

As commonplace as user generated content has become, it has not yet been realized to its fullest potential. In its purist form, the idea of user generated content offers limitless potential; allowing a game to be altered and expanded upon by anyone who has access to it, and allowing these changes or additions to be shared with other players. This would potentially create a self-perpetuating cycle of infinite expansion and replayability. In its current state however, user generated content falls short of this vision due in large part to a problem with its implementation. The root of this problem stems largely from the fact that the act of making a game is far less fun, and therefore far less appealing than playing a game. Few would prefer spending hours clicking through menus and moving objects around a 3D viewport to the more visceral and immediately rewarding experience of playing a game.

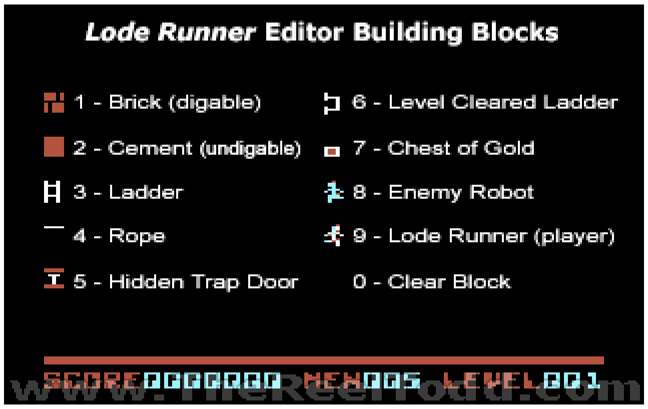
This thesis aims to propose that this discrepancy between playing the game and creating content for the game can be resolved by making the process of content creation more closely tied to gameplay itself. If the core of a game revolves around the creation of content, be it characters or levels, then new content becomes a natural byproduct of playing the game. This allows players to create environments by playing the game rather than forcing them to learn a new skillset and engage in a creation process that is not in itself immediately fun or engaging. In addition, the creation of content through gameplay gives players goals, risks and rewards that both encourage creation and affect the created content.

2. User Generated Content – A Historical Perspective

Plenty of today's games are developed with user generated content in mind, but a few short years ago that wasn't the case. This is not however to say that user generated content is anything particularly new to games. Indeed, it has existed as an established part of gaming for a very long time. Some of the earliest examples of games that rely on players developing their own scenarios and stories can be found in the early 20th century with the first published set of rules for games played with miniature figures. The novelist H.G. Wells published two very simple rule books for miniatures, *Little Wars* in 1911 and *Floor Games* in 1913 (Costikyan). By providing players with a simple set of guidelines to authoritatively determine the outcomes of battles, these early miniature games put most of the creative responsibility on the players whose scenarios and troop placements brought the systems to life. Players of this game could be considered some of the first users to create their own levels. It is clear that players enjoyed this creative freedom, as miniature games are still thriving today. Modern miniature games allow players to not only create battles, but also armies, with players spending countless hours collecting and painting each individual figure, putting no small amount of time and creativity into their units and armies.

Along with miniature games, other forms of tabletop games also rely on players' creativity. 1973 saw the release of the first edition of Dungeons and Dragons (Costikyan). This hugely successful franchise, while offering numerous volumes of material detailing characters, items and locations, relies heavily on players to develop their own content and shape their own experiences. The controlling player, known as the "dungeon master," has a very active role in defining the experience for the other players. The vast majority of dungeon masters create their own levels for their group of players to experience, drawing areas out on grids and arranging enemies and obstacles.

An argument can be made that players of non-digital games are naturally closer to content creators than those of digital games. However user generated content has been present in digital games for quite a few years as well. One of the earliest games to allow users to create their own levels was 1983's Lode Runner. Created by Douglass E. Smith at the University of Washington, Lode Runner was a puzzle platformer that challenged players to move around different levels, avoiding obstacles and collecting gold (Casmassina). One of the games defining features, contributing greatly to its success was the game's included level editor. The game was a huge success. Even early prototypes were very popular among students at the University of Washington. Smith noted that "It was a game that was available to all the students and it became kind of a cult thing. It got put up there were everyone had access to it and a lot of people started playing it. Part of the appeal was that people would design levels, and I'd try to add new features (Casmassina)." The level editor included with Lode Runner was fairly simple. Editing levels was done by manipulating a curser around a grid like screen with the arrow keys and placing different blocks and level pieces (Washburn). These levels could then be saved and shared with others. Figure 1 shows the different blocks that users could use to design their levels.



Lode Runner Editor Building Blocks

1 - Brick (digable)
2 - Cement (undigable)
3 - Ladder
4 - Rope
5 - Hidden Trap Door
6 - Level Cleared Ladder
7 - Chest of Gold
8 - Enemy Robot
9 - Lode Runner (player)
0 - Clear Block

SCORE0000000 MEN005 LEVEL001

www.TheReelTodd.com

Fig. 1. An Image of the Lode Runner Editor; "Lode Runner Editor"; *Lode Runner Level Editor Tutorial*; TheRealTodd.com, 5 Aug. 2008. Web. 16 Apr. 2013. <http://www.thereeltodd.com/img44x0123/LodeRunner_editor_key.gif>

Lode Runner was quite popular in the U.S., but it saw even greater success when it was released on the Famicom in Japan (Casmassina). Perhaps the games popularity and the novelty of the level editor contributed to another early game to feature user created levels, *Excitebike*. This Famicom game was released in 1984 in Japan, and was a launch title for the Nintendo Entertainment System a year later in North America. This motorcycle racing game allowed players to place ramps, jumps and hazards on their own tracks which they could then race through. This creation mode was limited however, in that the game cartridge did not include a battery for saving data, meaning that any created tracks were lost when the system was turned off (Harris).

Both *Lode Runner* and *Excitebike* show that developers in the early 80s already recognized the appeal of giving users the ability to create their own levels. With other games however, users found a way to modify games without the knowledge or help of the developers. In the 80s and early 90s a new genre of games was emerging that would quickly rise in popularity. Wolfenstein 3D and its successors were among the first games to have their functionality modified by consumers, a process that came to be known as modding. In 1983, two programmers, Andrew Johnson and Preston Nevins, modified *Castle Wolfenstein, replacing* the Nazi enemies with Smurfs. Figure 2 shows the modified title screen for *Castle* Smurfenstein. This is often credited with being the first mod of a commercial game, and its creation process was nothing short of hacking the game. Nevins recalls the process, saying, "We learned that Wolfenstein's graphics were stored as a software font, and just 'printed' to the HGR screen. A bit of pixel twiddling, and the Nazis became (ugly, skinny) smurfs with guns (Nevins)." This marked the beginning of computer game modding, a practice that would become fairly common among computer gamers and hobbyist programmers. *Wolfenstein 3D* and *Doom* saw numerous modifications both by amateurs and professionals alike. These mods quickly grew in popularity, eventually catching the attention of the games' developers, who eventually embraced the practice. Shawn Green of Id Software commented on the company's stance, saying

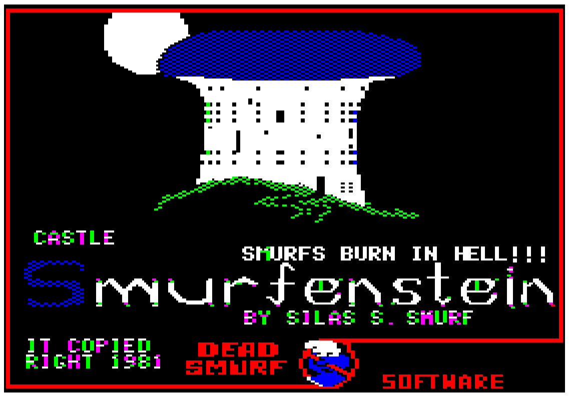


Fig. 2. The Title Screen for *Castle Smurfenstein*; "*Castle Smurfenstein*"; The first 'Official' Castle Smurfenstein Home Page; Dead Smurf Software, Sept. 1999. Web. 20 Aug. 2013. <http://cvnweb.bai.ne.jp/~preston//other/deadsmurf/index.html>.

> "We do not detest it at all. Actually, at one point, we released code for correctly compiling DOOM maps. John Romero and American McGee also subscribe and frequently post to an internet mailing list about the creating on DOOM levels…Our general opinion is user created levels are good for users and the game as well. The only objection we have is the use of the levels with the shareware version and the selling of levels (Lockwood)."

As this phenomenon continued and many more mods were created, Id realized the potential offered by embracing mods. Since mods require a copy of the original software to run they didn't detract from a game's potential sales, and in fact contributed to the potential success of a game, creating more buzz and prolonging players' enjoyment of a game. To help nurture this newly created community, Id shipped their future titles with creation tools that helped players design and implement their own levels (Kücklich). Throughout the later 1990s and 2000s modding tools became fairly common among computer games. While many first person shooters included modding tools, plenty of other genres made them available as well. Real-time strategy games such as *Civilization* and role-playing games such as *Neverwinter Nights* and *The Elder Scrolls* also embraced the idea of user generated content. Developers started harnessing the power of allowing players to modify their game and modding became an increasingly important feature to add to a game. Todd Howard of Bethesda Softworks epitomized the attitude of many developers during this period by saying of modders "It's their game. They paid for it. Let them do what they want (Gamespot)."

While user generated content has enjoyed a thriving life among PC gamers through the modding community, it had a much smaller presence on consoles until fairly recently. Throughout the 90s and early 2000s there were only a few console games that allowed players to create their own content. This disparity was most likely do to hardware limitations, lack of connectivity and the generally more closed ecosystems found with consoles at the time. With the start of the current console generation in 2006 these restrictions began to change. The increased implementation of online features and increased amount of local storage on these newer consoles offered developers the opportunity to open up their games to users' creativity. With this new emphasis on user generated content however, there have been only a few standout games that used the feature in a truly novel or innovative way.

3. Notable Recent Implementations of User Generated Content

One of the most celebrated and praised games released this generation that offered players the chance to create their own levels was Sony and Media Molecule's *LittleBigPlanet*. This side-scrolling platfomer gave players the ability to create and share original levels through the use of in game creation tools. These tools offered players the ability to place pre-built objects or create their own, either by carving out their own shapes with simple primitives, utilizing existing components like engines and strings or gluing various components together (Viso Games). *LittleBigPlanet*'s edit mode can be seen in Figure 3. These tools, while simple to use, allowed players a great deal of freedom to create surprisingly complex objects and mechanisms. An important element of this system, and one that contributed to its novelty and success was the ability not only to save and distribute completed levels, but also to share any objects a user had created. These objects could then be used by others in their own levels, which resulted in many new and creative mechanics being utilized in user

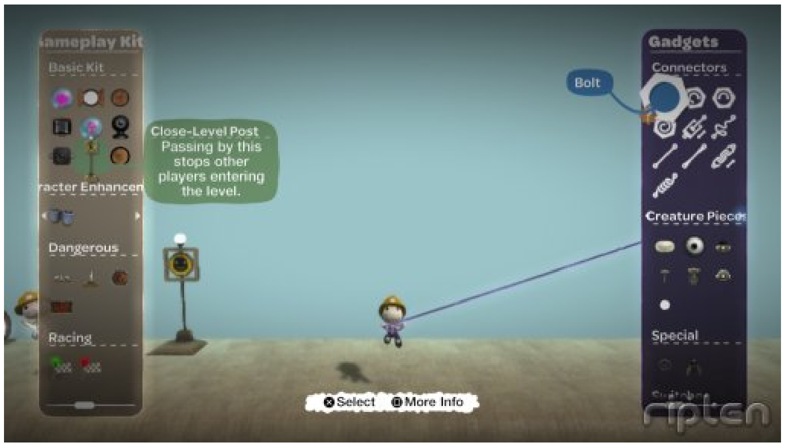


Fig. 3. *LittleBigPlanet*'s 'Create Mode;' "LittleBigPlanet Creation Mode;" *LittleBigPlanet Review*, Ripten.com, 4 Nov. 2008. Web. 21 Aug. 2013.

created levels (Robinson). Sharing levels and objects was made exceedingly simple in *LittleBigPlanet*, and a thriving community quickly grew around the game and its numerous sequels.

Another game produced by Sony that featured user generated content was 2010's *Modnation Racers*. This traditional kart racer featured a track creator that was very simple to learn and use. Using this creator, players were able to design tracks with minimal effort. By making the creation tools similar to the game's standard driving controls, *Modnation Racers* was able to make track creation easy for players who were familiar with how the game worked. To create a track in *Modnation Racers,* players controled a vehicle similar to a paving machine. Wherever the player drove, the track was created. To further customize their track, players used a myriad of tools to paint trees, sculpt hills, and place various other walls and pieces of scenery. There was also a tool that players could activate that would automatically complete a track, generating the remaining section and linking back up with the starting line (Ho).

Another franchise that has not only embraced user generated content, but has also attempted to make the content creation process more closely resemble the gameplay is Microsoft's *Halo* series. The third installment, released in 2007, introduced a creation tool called Forge Mode to the series. In this mode players had the power to transform their characters into a freely floating orb with the ability to create and position objects, effectively allowing players to create their own multiplayer maps. As with *LittleBigPlanet*, *Halo 3*'s Forge Mode attempted to make a level creation tool that resembled the core gameplay experience as closely as possible. Players could play Forge Mode online with friends, and the game even kept a score similar to other online game modes (Timmins). This led to all sorts of emergent gameplay mechanics cropping up as players learned to splatter each other while placing objects in the level. Thousands of online maps were created and shared using these tools, and new game modes imagined by players were later fully implemented into the game.

4. Benefits of User Generated Content

One of the chief benefits of allowing users to create content, from a developer's standpoint, is that it allows for vast quantities of content to be produced for a title, without overloading developers. Level designers do not have to create countless levels if players can create their own. Game designer and author Rudolf Kremers recognizes this benefit, noting that "A game that allows players to manufacture their own gameplay experience produces an added bonus of *free* content. After all, if *automatic* generation of gameplay occurs, it does not require work from the level designer. It is gameplay content created by somebody else, which in the busy schedule of professional level design is a very good development (Kremers)." Level design is a very important part of game development. A game's levels often define the gameplay experience more than any other element. In spite of its importance, level designers can only be expected to develop a finite number of levels before the game must ship, but if users are allowed to create their own, these core levels can be augmented with countless others.

Developers have realized the many benefits of allowing users to create their own content, but why do players who might otherwise be playing a game, instead spend their time toiling away with an editor creating content for others. Many players are passing up the immediate satisfaction of playing a game, for creating content, a process offering no perceivable immediate rewards. This decision has been written about by many scholars, including Clay Shirky, who asks the question "Of all the things to do online, what would motivate someone to give up this much of her own time and money for something that produces no obvious tangible reward (Shirky)?"

One of the chief motivators for content creators comes not from external rewards for creating content, but from the act of creation itself. The creation process offers creators intrinsic benefits that motivate them to keep creating. For these individuals, creating is its own reward. This applies not only to games, but also to many different outlets that can be found on the internet. Within the past ten years, online users have shifted from purely consuming the available media to producing media of their own and sharing it with others. YouTube, Flickr, Tumblr, DeviantART, Vine and countless others have seen astronomical success due to contributions from millions of users around the world. These users are choosing to spend hours of their precious time creating content to share with others. This is, for the most part, done not for personal gain, but purely for the enjoyment of creating and sharing.

Though the process of creating may be sufficient to motivate a fair number of creators, there are some extrinsic rewards that emerge naturally as a result of sharing content. Online communities and YouTube celebrities are proof that content released on the internet can foster appreciation and feedback from others and can offer rewards such as fame, notoriety or even financial support to content creators. Sharing is in fact a key component of content creation. Whether users create images, music, videos or content for a game, sharing is most often the goal of creating something. When discussing the online trend known as lolcats, Shirky describes the desire to share content, saying "The phrase 'user-generated content,' the current label for creative acts by amateurs, really describes not just personal but also social acts. Lolcats aren't just user-generated, they are user-shared. The sharing in fact, is what makes the making fun—no one would create a lolcat to keep for themselves (Shirky)." Shirky may describe lolcats as "the stupidest possible creative act, (Shirky)" but the desire to share lolcats is the same desire to share other creative works, including content and levels for games.

Game content is no exception to the social nature of creating. There are many websites and communities dedicated to the creation and sharing of content for games. Valve's digital game distribution service Steam has been praised for creating a large community surrounding PC gaming. A major part of this community is a built in platform for sharing user generated content called the Steam Workshop. Through the Steam Workshop, users can share content for over fifty different games. In addition to this, many games that feature user generated content also have websites centered around sharing mods. The Nexus network of sites offers mods for many different games. SkyrimNexus, the site devoted to mods for The Elder Scrolls V: Skyrim, is among the largest, and offers over 28,000 different mods for the game.

For every mod that is created and shared, through the Steam Workshop or through other means, there is almost always a means for others to comment on the work, by leaving a comment, through a forum thread, or simply by giving the mod a rating. This feedback system gives users who choose to experience the mod the opportunity to interact with the mod's creator, giving him or her feedback or suggestions for improving the content. This exchange allows ideas to spread and communities to thrive. Kareem Ettouney, the art director of *LittleBigPlanet* recognized how crucial these communities are, saying "The most important thing about *LittleBigPlanet* is not that it does blockbuster sales, but that it survives. We want it to be around, and we want the community to stay there, and as long as they're there that's when it grows (Robinson)." These comments and suggestions often directly influence the work, as creators continue to improve their mods and release new versions. This dialogue between creators and consumers also finds its way back to the game's developers. Often a feature introduced through a mod will end up being officially implemented in the game either through an update or through a future release in the series. In her paper on online FPS games and their communities, Sue Morris proposes that such games could be identified as "co-creative" media, saying "I would argue that multiplayer FPS games are "co-creative media"; neither developers nor player-creators can be solely responsible for production of the final assemblage regarded as "the game", it requires the input of both (Morris)." These types of games in particular are likely to see updates and new modes or features created in response to the community.

5. Issues with User Generated Content

The value that user generated content adds to a product is clearly demonstrated by the wealth of titles now offering the feature on both consoles and PCs. Though the volume of titles has increased in recent years, the feature still suffers from overly complicated implementations that require more skill than most players possess, and more time to learn than most players are willing to invest. Most players would much rather simply play their games. As rewarding as creating content for others can be, most people would agree that consuming content is much more immediately enjoyable than creating content. Results gathered from recent games shows that the vast majority of game players feel this way too. In a recent article about this very problem, Scott Sharkey noted that

> "Take, for example, the number of copies of Halo 3 that have been sold against the number of maps users have created in Forge. At the moment there are about 20,000 custom maps online. It seems like an awful lot, but bear in mind that this is a game that has sold over 8 million copies. That's one map made for every 400 copies of the game floating around out there. Even disregarding the fact that most of the users who make any maps at all will tend to make more than one, it's clear that only a fraction of one percent of the people who've bought the game are bothering with the level editor (Sharkey)."

In many cases, developers attempt to win over more players by creating simpler editors that users can pick up and learn with little time investment. This is only a partial solution however, as even a simple level editor is still a new set of tools the user has to learn. The user must learn how to utilize the different functions, and how the various components can be used and combined to create a level. *Skyrim's Creation Kit*, as shown in Figure 4, is about as complex as an editor can be. Moreover, users must learn a set of skills and rules that go along with creating a playable and fun level. As easy to learn

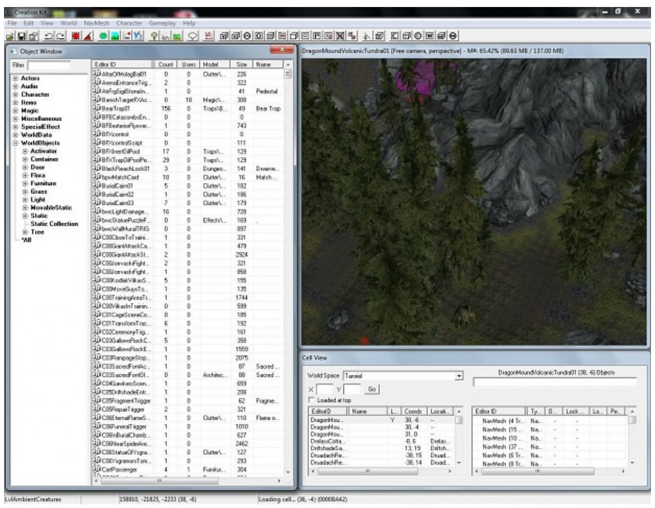


Fig. 4. A Screenshot from the Skyrim Creation Kit; "Skyrim Creation Kit;" *The Creation Kit Has Been Released*; *Moddb.com*, 7 Feb. 2011. Web. 24 Aug. 2013. <http://media.moddb.com/cache/images/members/1/386/385498/thumb_620x2000/CreationKitFirstPic.1.jpg>.

as an editing tool may be, the vast majority of users will still forgo learning any tools and go on enjoying the experience of playing their game. In order for user generated content to truly change the percentage of players that contribute content must greatly increase. For this change to happen, our way of approaching user generated content must also change.

6. Procedural Content Generation

Procedural content generation is a practice employed by many games as a way of automating tasks that would otherwise cost time and manpower to carry out by hand. By programming an algorithm into the game, new content can be randomly created on the fly rather than being designed by hand. This approach can work for environments, levels, characters or items. By programming simple rules for generation, limitless content can be created without relying on designers or artists.

One of the earliest games to utilize procedural generation was the 1980 fantasy RGP *Rogue*. This game was able to randomly assemble rooms into a maze like dungeon and fill them with enemies and items (Lambe). Other games followed up on this trend and applied user generation to create their own levels. Space exploration games including *Starflight and Elite* were able to procedurally create entire galaxies for players to explore (Lambe). Since then procedural content has found its way into many games. Often it is used not to create entire levels, but rather as a tool for rapidly placing objects that can then be moved around and tweaked by hand to create the final level. One such instance was during the development of the base jumping game *Aaaaa!.* The developers created methods for procedurally placing goals throughout their levels in order to speed up production and create creative algorithmic patterns (Lambe).

7. A Unified Model for User Generated Content

For a greater percentage of players to contribute to content creation, a new approach to content generation should be taken. If user generated content is approached not as a separate process the user must learn, but as a natural byproduct of playing the game, everyone who plays the game will consequentially be contributing content. Even the simplest of level editors requires the user to learn a new skill, but if the skills the user learns while playing the game are the same skills he or she uses to create content, creating will be just as enjoyable as playing the game. Moreover, if the content is created during the gameplay process, the player will contribute content every time the game is played. While creating a level for a game can be a rewarding process, it can take months of tedious work. As an alternative to this, levels can be created dynamically as the user is playing the game. This creation process can be designed to react to the user's actions, allowing users to create the level they desire by behaving a certain way as they play. This reactively generated content would essentially merge the creation process and the consumption process. Creating levels would therefore be the same as playing the game.

This approach would not only expand the pool of content creators, but also change the way content is created. Creation becomes not a process for the few who possess the artistic and technical skills, but a part of the game itself – a byproduct of the user making progress in the game. Many players shy away from creating content in favor of playing a game, but by harnessing players' desire to play and using that desire to fuel the creation process, content creation can be made much more appealing, and much more fun.

8. Similar Approaches Currently in Use

Though by and large the content creation process has not yet come to embrace this method, there have been a few notable examples of games that take a similar approach. There are several games in which creating content is essential to the gameplay experience. One of the games to take this approach was 2008's *Spore*. In *Spore,* players start out as a microscopic creature and follow it as it evolves into progressively more complex creatures, eventually becoming sentient and forming a society before finally embarking across the universe and colonizing other planets. As stated before, creating content for the game is an essential part of the gameplay. Each time the player's creature gains enough experience to evolve, the player must evolve his or her new creature, choosing from a number of body parts, combining and modifying them to create the resulting animal. After the creature becomes intelligent enough, buildings and vehicles are created in the same way. The most innovative and visionary element of this system is that once a player creates a creature, building or vehicle, these creations are uploaded to a server, and subsequently downloaded and used to populate other players' games. This created a platform for players to craft their entire experience while playing the game. When describing *Spore* at a TED talk the game's creator, Will Wright, noted that he wanted players to be "building this world from their imagination," and he wanted the game to be "extracting it from them with the least amount of pain (Wright)." Though it still relied heavily on the player to learn an editor, *Spore* clearly took steps towards exploring the greater potential of user generated content. Figure 5 shows a number of creatures created while playing Spore.

Fig. 5. A Screenshot from *Spore*; "A *Spore* Screenshot," *Creatures*; *About.com*. 28 May, 2008. Web. 4 Nov. 2013. < http://compsimgames.about.com/od/spore/ig/-Spore--Screenshots/Spore---Creature-Stage.-0SL.htm>.

Another example of a game with a more integrated approach to user generated content is 2012's *Hitman Absolution*. The vast majority of this title does not feature user generated content, however there is a specific mode in the game that does. This mode, called Contracts Mode, offers players the ability to create their own missions and share them with other players. While this concept by itself is nothing new or surprising, the way in which it is implemented makes it a notable application of the unification of core gameplay and content creation. The beauty of Contracts mode is that it is nearly identical to the other modes of the game. The player's interactions with the world are preserved as well as game mechanics and presentation. The only difference is that instead of being given an explicit objective, players create their own objectives while playing. Contracts Mode drops players in a sandbox like environment. They are free to move around and interact as they wish. As the players interact, certain actions they perform are recorded, including anyone a player assassinates, the weapons he or she uses, and any disguises he or she chooses to wear (Blystad). These actions are then saved and become objectives for others to complete. Tore Blystad, the game's director described the feature, saying "…One thing that we did with Contracts was actually having the creator play the game as he creates the contract, and this kind of play to create mode as we call it is something that we did to avoid having a cumbersome editor or kind of a god mode or something (Blystad)." This implementation may be limited, comprising a smaller portion of a larger game, but it is an excellent demonstration of merging content creation with the core gameplay concept.

Another strong example of this concept is the astonishingly popular indie title *Minecraft*. Beyond any other title, this demonstrates the potential of user created content to generate interest, and thus revenue, for a game. *Minecraft*'s concept is simple. Players are dropped in a procedurally generated world made out of blocks. These blocks are made of different materials such as iron or rock. Players mine these blocks and use them to build structures, devices and other tools that help them dig deeper and find even more rare materials. Figure 6 shows the modified world created by a player of

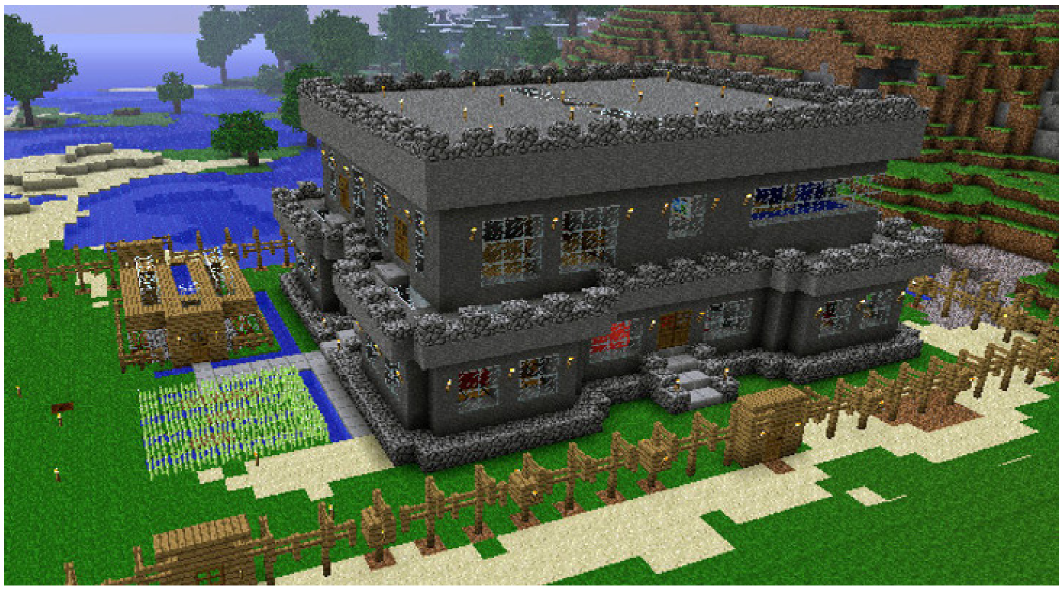


Fig. 6. A Building Created by a Player in *Minecraft*; "Minecraft Screenshot," Minecraft 1.5.2 is Officially Out; *Softpedia.com*, May 3, 2013. Web. 4 Nov. 2013. < http://news.softpedia.com/news/Minecraft-1-5-2-Is-Officially-Out-Download-Now-for-Mac-OS-X-350681.shtml>.

Minecraft. This ingenious process results in the player not only progressing through the game, but also shaping the world to his or her image. Moreover, with the ability to play with others online, each player is not only shaping the world for himself, but for others as well. It is impossible to play *Minecraft* without modifying the game world. The act of creating (and destroying) is inseparable from the core gameplay concept. Though it has a vague goal and there are steps that players must take to "progress," creation is really the only goal and motivation for players. The real beauty of *Minecraft* is that user generated content is not just a part of the game, it *is* the game. Creating in *Minecraft* is playing *Minecraft*. This surprisingly simple concept was novel enough to make *Minecraft* a huge success, and inspire a slew of games that shamelessly imitated *Minecraft*'s concept.

Other games have started appearing that are influenced by *Minecraft*, but instead of imitating the gameplay of *Minecraft* exactly, these new and upcoming games applied the same concept to a more original gameplay system. Microsoft is currently planning to launch its new console the *Xbox One*. One of the games being developed for the console is *Project Spark*. This game, like *Minecraft* is centered around player generated content. In a similar vein to *LittleBigPlanet*, *Project Spark* gives players access to a number of editors that allow them to create their own games, sculpting environments, placing and creating props, and creating their own gameplay by manipulating a simple logic system that can be attached to any object (Persson). Though *Project Spark* essentially presents players with an editor, this editor is really the main mechanic of the game rather than a separate mode included with an already built game. There is no other goal in *Project Spark* than creating, sharing and experiencing content. This pushes it more towards a toy than a true game, but it is still a noted step towards unifying the creation process with a game's core gameplay loop.

Sony has also realized the power of centering games around creation. During the development process of their upcoming online role-playing game *Everquest Next*, Sony Online Entertainment is leveraging the power of community driven creation by allowing players to participate in the creation process of *Everquest Next*'s game world. With a product called *Everquest Next: Landmark,* Sony is giving players a *Minecraft* style way to shape their virtual world. *Landmark* allows players to create their own landscapes by giving them a "plot" of land to shape as they desire. By creating and manipulating square units called voxels, players can shape the earth and create structures (Tuttle). As with *Minecraft* players must gather resources, however in *Landmark*, only the owner of a plot can modify it, unless he or she chooses to allow others to do so. Players can gain additional plots of land by playing the game, and are offered the opportunity for their plot to be used in the game. SOE is planning to hold competitions to see who can design the best section of land based on certain design and artistic criteria. The winning plots will be selected, and ultimately implemented in *Everquest Next* (Tuttle). As with *Project Spark*, *Everquest Next: Landmark* is centrally focused on the creation process. Unfortunately, it looks like players are still going to have to use an editor in this case, but by making the creation process the means of progression, and by offering payoffs and rewards for creating, *Landmark* has made everyone who plays the game a content creator.

It appears that developers are not only utilizing users to ease the load of developing large games, but are starting to realize the appeal for games centered around the creation process. These upcoming games are aiming to reward players for creating. They transform the creation process from a secondary activity for the few who possess the skills necessary and who appreciate its intrinsic properties, to the natural byproduct of gameplay. The seeds of this new type of content creation are starting to take root.

9. My Implementation: *Aeroblazer*

After examining user generated content, its uses, values and shortcomings, I developed a visual component to demonstrate an implementation of an improved content creation process. I needed this demonstration to meet a number of design goals. The visual component had to address the disparity between the number of players who experience a game and the number of players who create content for that game, remove the barrier of entry for potential content creators, unify the content creation process with the gameplay, so that playing the game results in content being generated, create a system that allows for a unique level to be created each time the game is played, and make the process of content creation more fun.

With these goals in mind, I created a prototype that aims to fulfill these goals and puts my ideas for unifying content creation and gameplay into practice. To achieve this, I chose to work with the Unity engine. Not only am I most familiar with this engine, but it also offers many capabilities that I knew I would need, such as a simple method for scripting mechanics and placing objects procedurally. In addition, Unity offers a fairly simple way to directly access 3D meshes, making it possible to modify the environment in real time. Using Unity, I designed and programmed a prototype racing game that allows players to create a track while competing against another player. This prototype, which I call *Aeroblazer*, uses procedural elements that react to players' actions to generate a track while they race.

Aeroblazer's gameplay takes the form of matches between two competing players. These players take turns creating and navigating a track that gets longer as the match progresses. A match starts out with the first player flying around a largely empty space, populated with scattered debris that must be avoided, pickups that can be collected, and a checkpoint that, if passed through, grants the player a point. While this player is flying around, a ribbon like track is being created behind them, following the path they took through the space. Figure 7 shows the track being created behind the top player. As the figure shows, a meter is

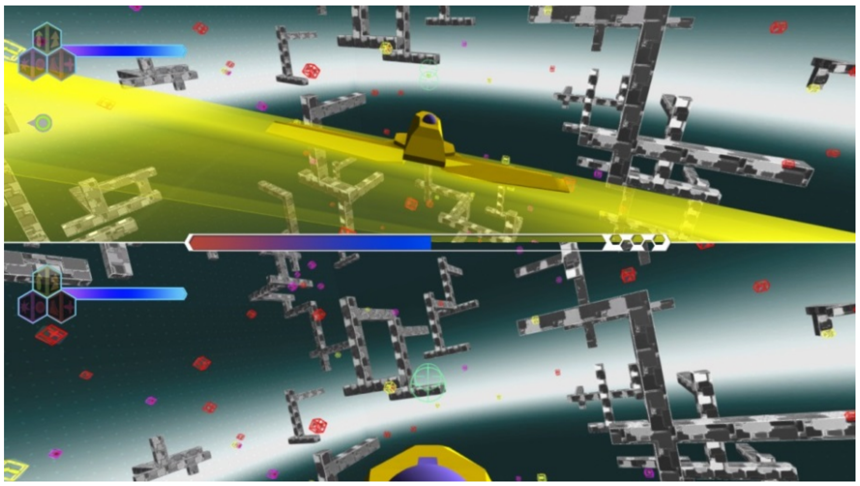


Fig. 7. A Typical Match in *Aeroblazer*

slowly emptying as the player is flying around. This represents the player's remaining flight time. When the meter reaches zero the player's time to fly and create is over. Once this happens it becomes the second player's turn. After this initial turn has ended, all subsequent turns begin with the active player having to navigate the track that has been created in all previous turns. Instead of controlling a plane, each turn now starts with the active player controlling a hovercraft like vehicle that races along the track as it exists to this point. If the player reaches the end of the track before time runs out, his or her vehicle transforms into a plane, and he or she can fly around, laying more of the track until the remaining time expires. While these two players are creating the track, they are also competing with each other. The edges of the track are lined with bumpers that, when hit, will bounce the player away from the edge, but will also take away a bit of his or her health. If one of the players' health reaches zero, the match is over and the other player wins. Alternatively, a player can win if he or she reaches the majority of five checkpoints floating around the creation space. The match only ends once all five checkpoints are reached, so getting three checkpoints does not guarantee a win. This can lead to a situation where one player's only chance for victory is to completely reduce the other player's health, resulting in more challenging and desperate sections of track. During this battle of creation, players can collect and use powerups that alter the track they are creating or make it easier to navigate. While one player is controlling his or her vehicle, the inactive player has control of a cannon mounted on top of the active player's vehicle. With this cannon, players can shoot floating powerups to collect them, and consequently keep the other player from touching them and thus collecting them for himself or herself. This mechanic leads to interesting situations in which the controlling player modifies his or her flight path in order to prevent the other player from collecting precious powerups.

This system allows players to create, while giving them goals, risks and rewards for doing so. Each time a match is played, a new track is created, making content creation the outcome of gameplay, and turning all players into content creators. Figure 8 shows a section of track that was the product of a match between two players.

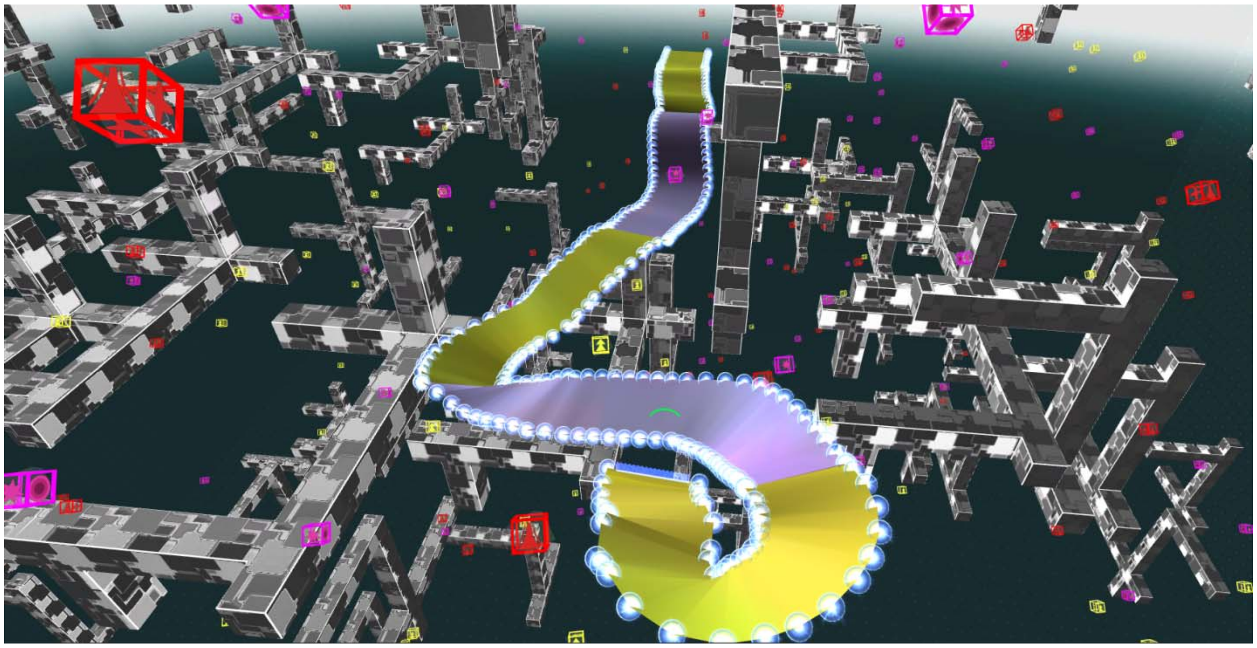


Fig. 8. Part of a Track Created as a Result of Playing *Aeroblazer*.

Through this system, the environment the users inhabit is largely a product of the decisions they make during the match. The choices players make as they avoid obstacles, collect and use powerups and fly through checkpoints have a direct influence on how challenging the course is for the next player, and consequentially for themselves on subsequent turns. This results in players trying to balance reaching goals, such as powerups or checkpoints, with creating a track that will cause trouble for the opponent. At the same time, this player will have to navigate the track on his or her next turn as well should the opponent survive. Players who do not have as much trouble with turns for example, might try to create a tightly winding course that will prove challenging for their opponent, but will not prove as hard for themselves. Powerups affect the track as well. Players can make an especially tricky section by creating a jump, making the track narrow, or creating a section where the bumpers lining the sides do increased damage if hit by a player.

This risk / reward system presents an interesting outcome. Unlike other methods of level creation, this system leads to levels that are the product of not only the user's imagination, but also the player's desire to achieve victory over his or her opponents. In fact the player's desire for victory is often the most influential factor in the final track design. The situations created during the match (sometimes literally) steer players as they create the track. This means that for a system like this to work, players will consequently have to give up some creative control over the final product.

10. User Testing

I tested *Aeroblazer* throughout its development. A number of different players played the game and gave very helpful feedback. Players generally enjoyed the game, and were able to grasp its concept relatively quickly. Players were quickly able to develop strategies for defeating other players, and the track designs reflected these different approaches. Sharp banking to rob the opposing player of a checkpoint, or devious use of powerups to create a devastatingly dangerous track segment resulted in some very diverse and challenging tracks. Users were definitely creating content, and they were doing it in ways than I had not imagined. Figure 9 shows several tracks created by players of *Aeroblazer*.

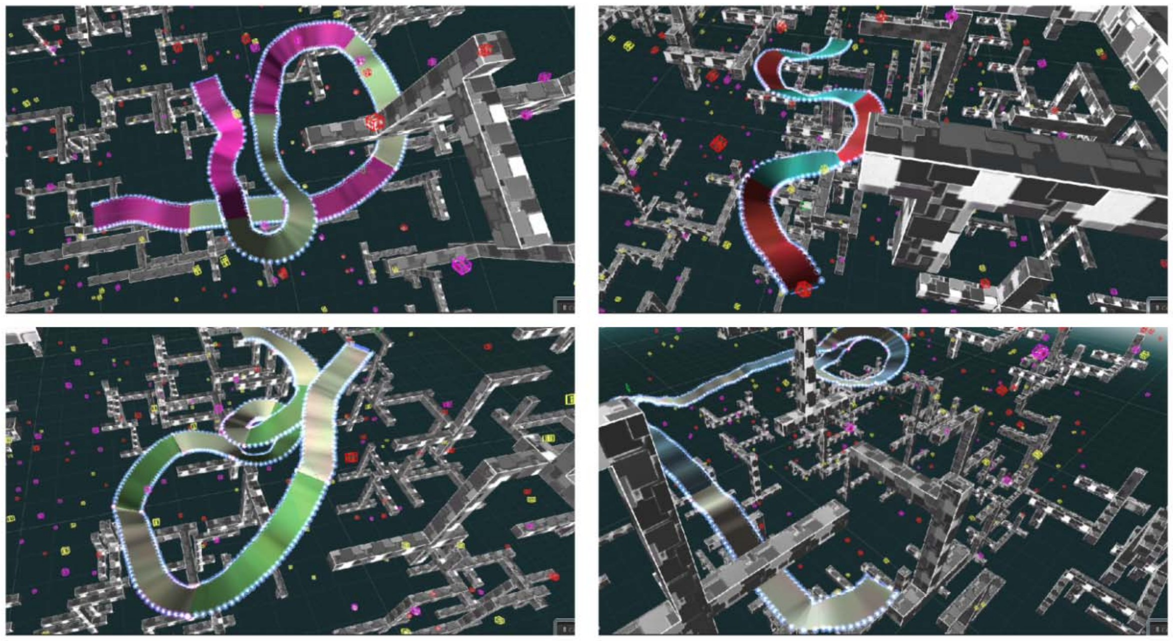


Fig. 9. Tracks Created While Playing *Aeroblazer*

11. Future Development

My implementation demonstrates many of the key elements of my argument. Despite being a functional and entertaining game, *Aeroblazer* could certainly benefit from further development to more completely illustrate this concept. There are a number of features that I plan to incorporate. There are a few mechanics in the game that need to be tweaked and fine-tuned in order to make the gameplay more fun. One of the most prevalent among these issues is the way the car handles. Players are able to navigate the track well enough, but I am not satisfied with the way driving the car feels. In addition, the camera following the car could be improved to make it easier for players to see the track ahead of them. Besides these implementation issues, *Aeroblaxer* could become even better with more development and additional functionality. The powerup system could be expanded with additional development. Allowing players to influence the track in different and more drastic ways would make each track more unique. In addition, a four player free-for-all mode would certainly inject additional variation into the experience, and affect the creation process in different ways.

Besides refining the visuals, tweaking the controls and adding new modes, there are a number of features that could be added or tweaked to make the gameplay and creation process more appealing. Chief among these improvements would be to add network functionality to the game. Like anything created today, the ability to share content with others is a major component of nurturing a community of content creators. Allowing users to share their created tracks or even encouraging content creation by offering rewards for sharing tracks would greatly increase the appeal and marketability of *Aeroblazer*. Exploring ways to gamify the sharing process, as well as encouraging collaboration on a larger scale would certainly energize the community and excite players to exchange levels, strategies, and experiences.

12.Conclusion

Through this implementation I demonstrated a content creation system that allows players to create levels while playing the game. While *Aeroblazer* successfully demonstrates the concept, it would need much more work to become a finished product, and could benefit from certain ideas being pushed further. Though I chose to implement this concept through a racing game, the idea can be implemented in many other genres, taking many other forms. The gameplay itself can take any form, but the creation process should happen at the same time. Imagine a 3D platformer game where the object is not to collect items scattered around the level, as it is with many platformers, but instead the player is given a limited amount of time to navigate the level, leaving items behind as he or she goes. Another player then must traverse the level finding the objects left by the first player. This mechanic can lead to players finding creative places to hide objects or using platforming skills to place items in hard to reach locations. In each instance, the initial player is creating a unique scenario for the second player every time. This concept could also be applied to a strategy game where one player builds a military base, placing buildings, troops and defenses while other players attempt to capture the base in the style of a first-person shooter. Almost any genre could benefit from this approach.

User generated content has grown to influence many games that are released today, but not all games are equal. While many users will always feel the need to create their own levels, characters and narratives, and to share those creations with others, many sill prefer consuming experiences designed by dedicated teams over many months. Not all games created in the future will allow users to generate their own content, nor should they, but those that do would do well to make the creation process as close to the gameplay as possible. Recent games like *Minecraft* and upcoming releases like *Project Spark* and *Everquest Next* are proof that the days of cumbersome and complex editors are coming to an end and the shift towards creating by playing has already begun. Developers are finding ever more creative ways of having players generate content, and as a result, more players are contributing content to the community—adding replayability and value to a game, and contributing to a better experience overall.

Works Cited

Blystad, Tore. *Hitman Absolution Contracts Mode Playthrough*. *Youtube.com*. Machinima.com, 28 Sept. 2012. Web. 3 Mar. 2013. <http://www.youtube.com/watch?v=5JcA0Ou1A8I>.

Game Director Tore Blystad walks viewers through the Contracts Mode in *Hitman Absolution*, describing and demonstrating the mode.

Casmassina, Matt, P. Schneider, and A. Devidas. "Lock'n'Lode." *IGN64*. IGN, 17 Feb. 1999. Web. 15 Mar. 2013. < http://www.ign.com/articles/1999/02/18/locknlode >.

IGN interviews Douglas Smith, the original creator of *Lode Runner* about the creation process as well as the then upcoming 3D version for the Nintendo 64.

Costikyan, Greg. "Don't Be a Vidiot." *Costik.com*. Greg Costikyan, 1999. Web. 20 Apr. 2013. <http://costik.com/vidiot.html>.

Greg Costikyan gaves a historical overview of non-digital gaming, including miniature games, wargames, board games and table-top games.

Gamespot. "Morrowind Editor Q&A." *Gamespot.com*. Gamespot, 8 June 2000. Web. 14 Apr. 2013. <http://www.gamespot.com/news/morrowind-editor-qanda-2585018>.

The Gamespot staff interview Todd Howard about the then upcoming Morrowind and its included modding toolkt. In it he discusses the company's rationel for releasing the toolkit and their viewpoint regarding mods.

Harris, Craig. "14. Excitebike." *IGN.com*. IGN Entertainment, 2011. Web. 9 May 2013. <http://www.ign.com/top-100-nes-games/14.html>.

An account of *Excitebike* as one of the top 100 NES games. The article is being used for its description of the game and of the creation mode therein.

Ho, William. *Modnation Racers - Track Creation*. Perf. William Ho. *Youtube*. United Front Games, 26 Mar. 2010. Web. 22 Sept. 2012. <http://www.youtube.com/watch?v=t8r8DWL9gj8>.

A video demonstration of the track creation tools in *Modnation Racers*. This video showcases the features and tools used to create tracks.

Kremers, Rudolf. *Level Design: Concept, Theory, and Practice*. Wellesley, MA: A.K. Peters, 2009. Ebook.

Rudolf Kremers, a veteran game and level designer discusses level design, analyzing concepts, creating a taxonomy, and citing examples to enlighten the reader on the process of successful level creation.

Kücklich, Julian. "Precarious Playbour: Modders and the Digital Games Industry."*The Fibreculture Journal* 5 (2005): n. pag. *Fibreculture Journal*. Fibreculturejournal.org, 2005. Web. 15 Sept. 2012. <http://five.fibreculturejournal.org/fcj-025-precarious-playbour-modders-and-the-digital-games-industry/>.

Julian Kücklich recounts the history and economy of modding and examines the legality and labor involved in modding, drawing comparisons and contrasts to things like open source development.

Lambe, Ichiro. "Procedural Content Generation: Thinking With Modules."*Gamasutra*. Gamasutra.com, 18 July 2012. Web. 14 Sept. 2012. <http://www.gamasutra.com/view/feature/174311/procedural_content_generation_.php#comments>.

Game designer Ichiro Lambe recounts his experience developing games using procedural generation. He highlights some strengths and weaknesses of working this way, and suggests using procedural content creation not as a replacement for hand placed content, but as an aide to creativity and a way of speeding up placement of objects.

Lockwood, Russ. "Doom Conference." *Doom Conference*. Computer Gaming World, Sept. 1994. Web. 14 Apr. 2013. <http://rome.ro/lee_killough/articles/doomconf.shtml>.

The transcript of an interview with Shawn Green and American McGee where they discuss Doom II, Doom Mods and Id's reaction to them.

Morris, Sue. "WADs, Bots and Mods: Multiplayer FPS Games as Co-creative Media." Thesis. University of Queensland, 2003. Level Up Conference Proceedings: 2003 Digital Games Research Association Conference, 2003. Web. 27 Oct. 2012. <http://www.digra.org/dl/db/05150.21522>.

A paper discussing the history of modding, as it pertains to FPS games. The author proposes that these types of games should be identified as co-creative media because of the way development incorporates input and advances made by the game's community.

Nevins, Preston. "The Other Dead Smurf Software Page." Dead Smurf Software, 14 Sept. 1999. Web. 20 Aug. 2013. <http://cvnweb.bai.ne.jp/~preston//other/deadsmurf/index.html>.

Preston Nevins, one of the creators of *Castle Smurfenstein*, considered the first game mod discusses the process he and Andrew Johnson went through to create the mod.

Persson, Saxs and Jerome, Claude. *Project Spark Gameplay Demo*. *IGN.com*. IGN, June 2013. Web. 10 Aug. 2013. <http://www.ign.com/videos/2013/06/11/project-spark-gameplay-demo-ign-live-e3-2013>.

A demonstration of Project Spark done by two of the game's developers. They show off the game, demonstrating a variety of its features.

Robinson, Martin. "LittleBitPlanet Interview." *IGN.com*. IGN, 24 Dec. 2008. Web. 20 July 2013. <http://www.ign.com/articles/2008/12/24/littlebigplanet-interview?page=3>.

Martin Robinson interviews staff members from Media Molecule, creators of the game *LittleBigPlanet*. The interview discusses various elements of the game, including user generated content and the community surrounding it.

Sharkey, Scott. "Why User Generated Content Has Failed to Change The Face of Gaming." Editorial. *1Up.com*. 1Up.com, 10 Mar. 2009. Web. 18 Sept. 2012. <http://www.1up.com/features/user-generated-content-failed-change>.

Scott Sharkey takes a look at the problem with user generated content to this point, and why it has failed to change the gaming landscape as much as it could.

Shirky Clay Cognitive Surplus Creativity and Generosity in a Connected Age New York Penguin 2010. Ebook.

Clay Shirky's main thesis is that people today have a Cognitive Surplus that they spend collaborating and sharing creations He makes note of several collaborative communities ranging in subjects from LOLCats to violence prevention communities

Timmins, Luke. "Bungie Weekly Update 08/03/07." Bungie.net : News. Bungie.net, 03 Aug. 2007. Web. 12 Sept. 2013. <http://halo.bungie.net/news/content.aspx?type=topnews>.

A news article from Bungie's website written before Halo 3 was released. It describes some of the capabilities of the game's Forge Mode.

Tuttle, Mark, John Smedley, and Dave Georgeson. "EverQuest Next Worldwide Debut."*YouTube.com*. Sony Online Entertainment, 02 Aug. 2013. Web. 28 Aug. 2013. <http://www.youtube.com/watch?v=Q4nswWwgLEw>.

The on stage reveal of Everquest Next and Everquest Landmark. The development leads show off the features of the upcoming games.

Viso Games. *Little Big Planet - Make A Level*. *YouTube*. Viso, 11 Feb. 2009. Web. 2 Aug. 2013. <https://www.youtube.com/watch?v=_vHdU9ctp7M>.

A brisk video demonstrating some of the features of the LittleBigPlanet Creation tools.

Washburn, Todd. "Lode Runner Level Editor Tutorial." *TheReelTodd.com*. N.p., 5 Aug. 2008. Web. 16 Apr. 2013. < http://www.thereeltodd.com/2008/08/lode-runner-level-editor-tutor.html >.

Todd Washburn describes how to create levels using the C64 version of Lode Runner. This article contains details on how the *Lode Runner* level editor works.

Wright, Will. "Will Wright: Spore, Birth of a Game." *TED.com*. TED Talks, July 2007. Web. 02 Aug. 2013. <http://www.ted.com/talks/will_wright_makes_toys_that_make_worlds.html>.

Will Wright describes *Spore* at a TED talk. He describes the game's mechanics as well as the experience it provides and his rationale behind creating it.