# Alternative Trajectories

## Structuring play through videogame physics engines

CONOR McKEOWN

A focus on technology is rarely a definitive characteristic of game studies. Throughout the familiar ludological/narratological dialogue that characterized the early field, a technological focus has largely been overlooked. Although the Copenhagen ‘school’ of game studies, particularly Jesper Juul, struggled to differentiate videogames from other media forms, those authors avoided rigorous technological analysis (Juul 2003). Recently, a body of scholarship has emerged, focusing on the specifics of game technology, of both hardware and software. Of note, Ian Bogost and Nick Montfort (2009), by reverse engineering the Atari VCS, have posited the effect of production practices on gaming aesthetics and play cultures. Montfort, with Mia Consalvo (2012) looked to the hardware of the Sega Dreamcast seeking to grasp its cult appeal. Regarding software, Montfort, Bogost and a host of contributors generated the code-focused *10 PRINT* (2013), examining the potential of a single line of code. In addition, Darius Kazemi (2014) draws our attention to the importance of studying source code for cultural analysis of games. There remains, however, a next step to be taken. Leading on from detailed studies of games as both narrative texts and aesthetic objects, a novel intervention in game studies would add the following field of observation: gaming technology. This approach would enable an insight into the immediate effect of algorithms on play and gaming practices.

In an attempt to combine code/hardware studies with rigorous analysis of gameplay I will analyse the impact of the ‘physics engine’ – the body of code responsible for the in-game representation of physics – on the first-person shooter (FPS) videogame, *Call of Duty: Ghosts* (2013). This essay concludes that physics algorithms, in FPS games in particular, shape the actions of players through two major methods: via the specific mediation of physical forces (such as gravity and drag) and the related modelling of object interactions. Focusing on the outcomes of these mediations we can begin to think of play practices as simultaneous productions shared between machine and human, drawing on conceptions of interaction familiar to media theory. For instance, as Timothy Barker states, ‘In any form of computer interaction, the user must operate within the limitations, function, and potentialities of the software.… By the same token, the software operates with the user’s capacities’ (2012: 115). Physics algorithms, it will be shown, have both shaped existing gaming practices and encouraged new ones.

The physics engine illustrates the dichotomy of digital games as simultaneously play objects and software systems. Although videogame physics shape the way we play, they are undeniably assemblages of digital code. They are perhaps best understood through the function they perform. A physics engine enables digital objects in videogames to behave in a familiar manner to real-world objects: as solids, liquids or gasses beholden to gravity. Beyond this, physics engines often produce an exaggerated or even entirely unfamiliar physical systems. Ultimately, they allow for the meaningful interactions of objects in a videogame. However, the term is relatively recent, coming in to common usage in the 1990s and initially only within the fields of education and engineering (Baszucki 2011). However,

ISSN 1352-8165 print/1469-9990 online
© 2016 TAYLOR & FRANCIS

using the term retroactively and applying it to early games can provide digestible examples of game physics. For instance, the familiar falling blocks of *Tetris* (Pajitnov and Pohilko 1984) give the appearance of simulated gravity through their simple animation. Applying knowledge of code/hardware processes, we can recognize this as the output of a rendering algorithm, shifting the position of the blocks within the software system and so visibly on screen. Although contemporary games use advanced techniques, enabled through superior hardware, the principles remain the same: through the clever manipulation of graphical rendering techniques, computers can come to mimic the appearance of real-world physical forces.

Within game studies, to date, game physics have largely been overlooked and are most often discussed within education. Martijn Koops and Martijn Hoevenaar's (2012) study found that children using games had a better grasp of mathematical concepts than those who didn't. What's more, they found little difference between children who used the game with periods of reflection against those who did not. A similar study found comparable results when children used games to learn about electromagnetic charges (Anderson and Barnett 2013). These insights are valuable as they provide a clear indication that programming specific physics algorithms, particularly when attempting to communicate ideas relating to mathematics and physics, can impact upon players. Building on these observations, I wish to illustrate that the impact of game physics are not limited to educational means; rather, that the physics that underpin a specific game shape the player's potential experiences within that game.

Modern FPS games provide an excellent example of how specific mediations of physics structure and shape gameplay. For instance, the *Call of Duty* series prides itself on 'realism' (Thier 2015). In this context, 'realism' seems to be equated with a satisfyingly accurate portrayal of Newtonian physics. This simulation is achieved via the game's physics engine. As such, the more a game's physics engine provides a model of space consistent with classical mechanics, the more 'realistic' it is likely to be considered. The physics engine used for the *Call of Duty* series was developed in-house by the game development studio Infinity Ward and simply dubbed the 'IW Engine'. It has been used throughout the series and licensed for use in others. While similar to other commercial software, the *IW Engine* has several unique features: for example, recent *Call of Duty* games represent complex physical effects such as smoke rendering, wherein the digital smoke effects respond to the player avatar and other digital objects on screen. The complex *IW Engine* is an example of the limits of contemporary games technology. It is worth noting, however, that even this physics engine, praised for its realism, presents physical forces in an augmented and amplified manner. The specifics of this mediation, I will illustrate, have an indelible effect upon play practices, shaping and accentuating them.

It is possible to analyse the aforementioned play practices within FPS games due to the well-developed practice of recording and online sharing impressive in-game achievements. These recordings provide easily accessible examples of play that can be analysed for the purposes proposed here. One such example is the video 'Top Ten Call of Duty Ghosts Trick Shots' (2014), compiled by the Machinima organization (a group that support filmmaking within videogame worlds). This compilation discusses technically impressive gameplay feats from the popular *Call of Duty: Ghosts* (Infinity Ward 2013). In the second clip of the compilation a player known as Direct Tubes achieves a 'kill', eliminating another player, by throwing a knife across the game map. The impression is given, however, that the 'kill' was not intentional as the recording player cannot see their opponent. Instead, they rely solely on their in-game heads-up-display (HUD) for an indication of their enemy's location. The thrown knife flies in an arc, drifting through a small opening in a ruined building, bouncing on the surface of the building's interior floor, out through another small opening, bouncing again off the exterior ground of the game map before hitting and, ultimately, eliminating the other player. Given the lack of clear intention of the shot taken and the non-classical physics at play, it is possible to argue that the kill shown here is less the action of the player and more, as suggested at the beginning of this essay, a co-operative action between player and machine. By investigating the physics behind this play in a focused manner, I hope to make this relationship clear.

The physics coding of *Call of Duty: Ghosts* takes liberties with Newtonian physics. For example, the knife discussed above does not lose velocity, even though it bounces off the virtual ground. It still has enough momentum to collide, fatally, with the opposing player. It is quickly clear that 'gravity' has been greatly abstracted in this game. While it will not be possible to use the exact parameters of the *IW Engine* here, due to copyright restrictions, it is possible to simulate similar results within a different software environment. For example, a common example of a defined gravity algorithm takes the following form:

```
public class GravityClass : MonoBehaviour
{ void Gravity()
{ Physics.gravity = new Vector3(0, -1.0F, 0); }
```

The line that reads 'Physics.gravity = new Vector3(0, -1.0F, 0)' is of central importance here. This common coding of videogame gravity simply applies a function to an object that results in that object being rendered one unit of distance below its previous co-ordinate in the next frame (similar to the downward appearance of falling blocks in *Tetris*). Programmers most often directly apply 'gravity' in this manner, as a downward force of acceleration. Ian Millington notes, 'although the acceleration due to gravity on earth is about 10 m/s$^2$, this doesn't look very convincing on screen. Most developers use higher values such as 15 m/s$^2$ for shooters to 20 m/s$^2$, which is typical of driving games' (2007: 50). This rendering effect can, when combined with other, mediated forces such as thrust and acceleration, create convincing and – perhaps more importantly – pleasing simulations of gravitational force.

It is important to acknowledge the ramifications of this algorithmic paraphrasing of the complex gravitational force. As we know, 'gravity' is anything but a simple downward force. Newton's universal law of gravitation maintains that the resultant force of attraction between two objects is equal to the 'universal gravitational constant' times the mass of two objects divided by the distance between their centres. The simplistic videogame 'gravity' is amplified in order to add excitement or increase a sense of verisimilitude. Players interacting with *Call of Duty*, therefore, are not so much having their play influenced by a mediation of a natural force but, rather, having it shaped and structured by an algorithm. In much the same way as players accommodate for the accelerating downward motion of blocks in *Tetris* or the distance that characters can jump in platform games, FPS play is influenced through an experience-based knowledge of that game's physics. Although players may conceptualize their movements within a virtual world as 'theirs', they are always engaging with the digital forces that shape those actions. As such, we are left with little choice but to conceive of action in videogames as a process of construction between human and machine. To play the game as efficiently as the players in the compilation, aiming, shooting and killing are informed not only by what is visible to the player, but also by a knowledge of the algorithmically physical behaviour of the digital world in play. As such, play as we know it is reformed into a new practice of interactivity between algorithm and human, player and machine.

Beyond the values of physical forces, it is also clear that the manner in which digital objects interact with one another has also been altered, furthering the sense that interacting with *Call of Duty: Ghosts* is a process of co-operating with algorithms. Analysing the game play described above, the final collision of the knife and the immediate death of the enemy player make it clear that *Ghosts* uses a programming concept known as 'primitives'. A 'primitive' refers to when the modelled physical behaviour of an object is not informed by its visual aspects but rather by a simple, invisible, three-dimensional mesh. Digital objects can be thought of as empty boxes, that contain the various functions, data or properties that designers fill them with. Noah Wardrip-Fruin discusses the quality of digital objects, defining them by their 'fuzzy' relationship between data and processes (Wardrip-Fruin 2009: 7). This fuzziness becomes clear when, for instance, analysing the behaviour of our knife. Although the data associated with the knife shows players a familiar, military-appropriate implement, the processual primitive used for the knife appears to be something simplified, ignoring the complexity of aerodynamics. It is likely that the programmers have used a rudimentary 'sphere-mesh' around the knife providing stable and efficient hardware processing. This explains how the knife can bounce twice and travel in such an unwaveringly straight line. Although using primitives in this manner is common practice, it radically alters the way that the game should be played.

Although it is difficult to imagine the practices of algorithmic play, wherein the player and the machine work together to construct new play paradigms, I wish to suggest that such practices are already in effect. Amidst the myriad recording of impressive feats of FPS game-play, both in *Call of Duty: Ghosts* and beyond, play practices have come to light that can be repeated by numerous players but that demand a great deal of skill (specific to videogame play). Chief among these is the '360-no-scope' kill, in which a player – unsatisfied with the already ample challenge presented by the FPS game they are playing – jumps, turns 360 degrees and then fires a shot, killing their opponent in one hit. Many examples of this practice are visible in the video already alluded to. This practice is symbolic of the developing symbiotic play practices of modern players. Having covered the basic physical traits of *Call of Duty: Ghosts*, as visible from the behaviour of the knife projectile, it is established that the simulated gravity is a simplified but controlling force while objects react to one another in a simplified manner. As such, although the 360-no-scope practice is impressive, it should be understood as an expression of knowledge of the structure of the game's particular modelling of physics.

Drawing on software studies it is possible to suggest that algorithmically structuring player actions also ideologically amplifies them. Contrary to the hypotheses of Lawrence Lessig and Alexander Galloway, wherein user resistance opens up rigid protocol to accommodating 'people's real desires' (2004: 176), Wendy Chun proposes that '[u]nlike any other performative utterance, code, almost always does what it says because it needs no human acknowledgement' (2006: 60). In the case of the knife shot, and the 360-no-scope, the *IW Engine* visibly prioritizes kills over non-kills, by simplifying the complexity of the physical world. Although algorithms amplifying actions may heighten the 'fun' of human players by raising the stakes while playing, from an ideological perspective, the game restricts human agency to a binary rubric of kill or be killed, inciting paranoia and shaping desire, as suggested by Chun. This certainly agrees with a large corpus of critical thought within the ethics of game studies today. Within Patrick Crogan's study of videogames as 'the relationship between the vision of simulated execution of high-tech warfare in virtual spaces and that of the leading edge of real weapons systems' (2011: 2) there is a strong implication that the simulation of ballistic physics in videogames is, in some way, an inheritor and perpetuator of military ideology.

While Chun's and Crogan's theories continue to have particular relevance as first-person shooters gain popularity year on year, their research is perhaps speculative to a fault. Indeed, Chun contends that 'people's desires too are generated by the system' (2006: 71). As we have seen in the analysis section of this essay, play practices are undeniably informed and enhanced by algorithms; however, human play practices similarly amplify and even, arguably, transcend the original algorithmic shaping of this virtual world. As visible through the practice of '360-no-scope' behaviours, although the physics of the digital world assist the players' potential to kill, players will react to this, adding increasingly human elements to the human/machine assemblage of game play.

Ultimately, this technologically focused exploration of gameplay, accepting the necessity of player agency, has identified novel gameplay practices and the role of software in producing them. It identifies that videogame players can adapt to the specific coding of a virtual world, such as the particular parameters of the physics engine mentioned above. Players, by accepting the specifics of software systems, develop novel approaches to play. These practices are subsequently reclaimed by the players and made significant in their community. As such, this study draws attention to the potential for novel play practices in videogames, made possible only through a particular merging of player and technology.

## REFERENCES

Anderson, Janice L. and Mike Barnett (2013) 'Learning physics with digital game simulations in middle school science', *Journal of Science Education and Technology* 22(1): 914–26.

Barker, Timothy Scott (2012) *Time and the Digital: Connecting technology, aesthetics and a process philosophy of time*, New Hampshire: Dartmouth University Press.

Baszucki, David (2011) "A Brief History of Videogame Physics Engines", http://blog.roblox.com/2011/12/a-brief-history-of-physics-in-video-games/, accessed 14 January 2016.

Bogost, Ian and Nick Montfort (2009), *Racing the Beam: The Atari video computer system*, London: MIT Press.

Chun, Wendy Hui-Kyong (2006) *Control and Freedom: Power and paranoia in the age of fibre optics*, London: MIT Press.

Consalvo, Mia and Nick Montfort (2012) 'The dreamcast, console of the avant-garde', *Loading … The Journal of the Canadian Game Studies Association* 6(9): 82–99.

Crogan, Patrick (2011) *Gameplay Mode: War, simulation and technoculture*, Minnesota: University of Minnesota Press.

Galloway, Alexander (2004) *Protocol: How control exists after decentralisation*, London: MIT Press.

Infinity Ward (USA: 2013) *Call of Duty: Ghosts*, multi-platform videogame.

Juul, Jesper (2003) 'The game, the player, the world: Looking for a Heart of Gameness', in Marinka Copier and Joost Raessens (eds) *Level Up: Digital games research conference proceedings*, Utrecht: University of Utrecht Press, pp. 30 – 45,

Kazemi, Darius (2014) *Jagged Alliance 2*, Los Angeles, CA: Boss Fight Books.

Koops, Martijn and Martijn Hoevenaar (2012) 'Conceptual change during a serious game: Using a lemniscale model to compare strategies in a physics game', *Simulation & Gaming, Vol. 44* 4(1): 544–61.

Machinima organization (2013) 'Top ten Call of Duty Ghosts trick shots: Episode 4', YouTube, http://youtu.be/iiLT_fatlvo,1December, accessed 14 January 2016.

Millington, Ian (2007) *Game Physics Engine Development*, London: Elsevier.

Montfort, Nick, Patsy Baudoin, John Bell, Ian Bogost, Jeremy Douglass, Mark C. Marino, Michael Matea, Caesey Reas, Mark Sample and Noah Vawter (2013) *10 PRINT CHR$ (205.5+RND (1)); : GOTO 10*, London: MIT Press.

Pajitnov, Alexey and Vladimir Pokhilko (USSR: 1984), Tetris Electronica 60, Videogame.

Thier, Dave (2015) 'Call of Duty: Black Op's 3's new realistic difficulty level sounds punishing', www.forbes.com, 5 October, accessed 14 January 2016.

Wardrip-Fruin, Noah (2009) *Expressive Processing*, London: MIT Press.

Copyright of Performance Research is the property of Routledge and its content may not be copied or emailed to multiple sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.