Johanna Drucker in “Humanistic Theory and Digital Scholarship” articulates as a key transformation (and bone of contention) in the variegated and interdisciplinary terrains of the digital humanities, saying,

I am trying to call for a next phase of digital humanities [that] synthesize method and theory into ways of doing as thinking. […] The challenge is to shift humanistic study from attention to the effects of technology […] to a humanistically informed theory of the making of technology.¹

But what does it mean to do, to make? And what sorts of doing and making are privileged over others? In other words, what counts in this shift?

For some, digital humanities making comes down to code, programming, and working in the back end. Stephen Ramsay has famously provoked, “Do you have to know how to code? I’m a tenured professor of digital humanities and I say ‘yes.’ So if you come to my program, you’re going to have to learn to do that

Fig. 1. Opening screen. *Lim*. merritt kopas, 2012. Game still.

Fig. 2. Entering the first room. *Lim*. merritt kopas, 2012. Game still.
eventually.”\textsuperscript{2} Like Drucker, Ramsay argues that digital humanities “involves moving from reading and critiquing to building and making…Media studies, game studies, critical code studies, and various other disciplines have brought wonderful new things to humanistic study, but I will say (at my peril) that none of these represent as radical a shift as the move from reading to making.”\textsuperscript{3} It is this foregrounding of \textit{doing} and \textit{making} that I want to take up, think about, and tinker with, not necessarily to rehash old debates or to pick (or pit) sides. Rather, I hope to articulate alternative modes and forms of doing that engage the \textit{modus operandi} of making without depending on specialized or exclusionary barriers of entry. As Ramsay qualifies, “Personally, I think Digital Humanities is about building things. I’m willing to entertain highly expansive definitions of what it means to build something.”\textsuperscript{4} In other words, I hope to take advantage of and take for granted that doing, making, and building can and must include a range of practices, processes, and materialities, many of which are accessible, every day, even vernacular. Specifically, I want to argue that playing a digital game is critical doing, that playing \textit{is} making, and to embrace playing as making.

For example, I open a browser and enter a game’s \texttt{url}. The window starts #000000 black and a moment later, the game begins. merritt kopas’s \textit{Lim} opens with a pink background, the suggestion of walls made of dark gray squares, and the game’s title spelled out in burnt orange squares (see Figure 1). White text below offers terse directions and cautions. Using the arrow keys, I discover that my avatar is one of the orangey squares, which I slide around the 2D world and which is indeed bounded by walls, the walls of a maze. I glide around the walls, making a circuit of the initial “room.” I cannot move the other squares no matter how hard I press on an arrow key. An opening in the gray walls on the lower right-hand corner of the screen offers

\begin{footnotesize}
\begin{enumerate}
\item Stephen Ramsay, “Who’s In and Who’s Out?” blog post, 8 January 2011.
\item Ramsay, “Who’s In and Who’s Out?”
\end{enumerate}
\end{footnotesize}
the only direction and exit. At this point, as I experiment, pressing the Z key does nothing.

Even in the first few moments of *Lim*, the player has explored, discovered, and learned a great deal. Though they are not privy to the lines of code, the programming churning under the *mise-en-scène* of the game’s interface and aesthetics, the player is deducing, aggregating, experimenting with a kind of algorithmic sense, a procedural literacy. Movement, keystrokes, boundary, collision, direction, timing, sound, silence, color, darkness, and so forth are made intelligible through the player’s interaction with the game, with the code of the game. The player learns what is allowed by the game, and perhaps more importantly, what is not allowed by the game. It is this enactment, this negotiation, this relationship between player and game, action and algorithm that I want to define as a creative and constitutive act. It creates, it makes.

Alexander Galloway says, “If photographs are images, and films are moving images, then *video games are actions.*” He continues, “Without action, games remain only in the pages of an abstract rule book. Without the active participation of players and machines, video games exist only as static computer code. Video games come into being when the machine is powered up and the software is executed; they exist when enacted.” This coming-into-being through the cybernetic loop of player input and game feedback reveals that digital games are never disembodied, immaterial experiences.

In *Lim*, I maneuver my little square through the exit from the initial area, leading it through a narrow but crooked path. As soon as I leave the first room, a gray block appears blocking any retreat like a door or gate shutting behind me. My avatar starts to flash, change color, rapidly blinking red, bright green, blue, sky blue, pink, magenta, and purple. The narrow “hall” opens into a second room populated by a scattering of orange-brown

---

6 Ibid.
squares. As soon as my avatar enters the second area, its color stops changing and becomes burnt orange just like the other squares. Again, there is only one obvious exit. Once I leave the room, the exit is shut, and my avatar returns to its gleeful riot of colors. But the flashing colors stops as soon as I approach the third room. This time, my little square turns a dark blue. Three orangey squares live in the third room, and as soon as I enter, forcibly attack me, running into me like an offensive tackle. Every time I am hit, the screen shakes and there is a brash sound like a burst of static. Getting by the orange squares is a challenge; the pummel into my avatar knocking it in different directions. However, by holding down the Z key, my little square turns the same color as the other and the battering stops. While “blending” in, the game zooms into the avatar and movement is painstakingly slow. Lim’s renders the difficulty and toll of “passing,” of fitting in, of trying to be just like everyone else. By the fourth and subsequent rooms, the ability to blend is no guarantee of safe passage. Even while passing, the other blocks respond violently as if seeing through the performance and disguise.

The premise and mechanics of Lim are simple but elegant, and they straightforwardly demonstrate that games are simultaneously a “designed experience”7 and an “action-based medium.”7 The algorithm, the code of a game is executed, turned from information to machine states, from one electromagnetic form to another, from data to light, sound, traces on a hard disk, to vibrations in a game controller. Games are an “active medium […] whose very materiality moves and restructures itself — pixels turning on and off, bits shifting in hardware registers, disks spinning up and spinning down.”9 But algorithm and code also get transformed into raised heart rates, cramped fingers, sweaty palms, full bladders, strained eyes, curse words, competitive spirits, and piqued curiosities. As Galloway says, “One plays a

---

9 Ibid.
Fig. 3. Being attacked by unfriendly squares. *Lim.* merritt kopas, 2012. Game still.

Fig. 4. Glitch: escaping the maze. *Lim.* merritt kopas, 2012. Game still.
Playing as Making

And the software runs. The operator and the machine play the video game together, step by step, move by move. Here “the ‘work’ is not as solid or integral as in other media” but it still has some substance, some effect within and without the technologies of the game.10

Playing a game is ultimately about learning the rules, the affordances and limitations of the platform, interface, and program, about understanding not the code itself but sensing and manipulating the contours, the structures of the code. Playing reveals what the code is doing (or not doing) and understanding the totality of how each line, function, routine, and sets of programming work together. A piece of code never becomes fully legible and intelligible until it has run, failed, tested, revised, and run again. Though not unique to video games, in a deep sense, playing is never not playtesting; and playtesting is always the bringing into being the game, particularly for algorithmic objects. Running the code turns program to phosphorescent pixel, code to haptic controller, platform to player performance, machine potential to virtual reality. As Ian Bogost argues, “[a]rtifacts like [a video game] suggest that procedural literacy can be cultured not only through authorship, such as learning to program, but also through the consumption or enactment of procedural artifacts themselves. In other words, we can become procedurally literate through play.”11 Drucker extends this notion of procedural literacy by recognizing that code and algorithms are also “performative materiality,” which “shifts the emphasis from acknowledgement of and attention to material conditions and structures towards analysis of the production of a text, program, or other interpretative event. After all, no matter how detailed a description of material substrates or systems we have, their use is performative whether this is a reading by an individual, the processing of code, the transmission of signals through a system, the viewing of a film, performance of a play, or a musi-

10 Ibid., 2.
cal work and so on.” Here I would add playing a digital game. Much in the same way a reader must learn the interface of a book (something that has become naturalized), interfacing with a game requires navigating, parsing, and interacting with multiple layers of mediation and information, feedback loops of input, output, image, sound, text, and trial. Under the rubric of performative materiality, the “idea of a user-consumer is replaced by a maker-producer, a performer, whose performance changes the game.”

Returning to Lim, there is a curious and wonderful moment when the bullying blocks hit so hard and fast that they inadvertently knock my little square outside of the maze walls as if my 2D avatar somehow turned sideways, became a line, and slid through the thin, pink gap between blocks of gray. The world

---

13 Ibid., par. 36.
outside of the walls is wide-open and safe from all attack. In fact, I can maneuver up to the wall of the maze, close enough that some of the blocks sense my avatar and move to intercept, but they are trapped and unable to reach me. It is smooth sailing outside of the maze, though the pink expanse is still bounded. My avatar cannot fall off the map as it were. While free of the dangers and confines of the maze, my little square is also free to flash its multi-colored self. Eventually, if I follow the shape of the maze, I can reach the proscribed end of the game — another multi-hued square waiting at the end of the path. As I near the kindred square, even though the wall, the gameplay stops, both cubes blinking at one another, there are a few sounds like the thumping of a heartbeat, and the game ends and goes black.

There is no clearer moment of a game’s performative materiality then when the game fails, glitches, freezes, or otherwise breaks the illusion of seamless and invisible code, computation, and other machine processes. Glitches reveal the formality and fragility of the rules of the game and of the algorithm. Although merritt kopas has been decidedly cagey about whether or not the above glitch is intentional or not (if that is even possible), the ability to escape the maze and the “proper” or “correct” way to play and win the game marks alternative even radical opportunities in game play and design. They allow resistance and refusal of the “basic conclusion that to play in a digital sandbox one had to follow the rules of computation.”

This glitch in *Lim*, as I have written elsewhere, is a queer moment, a queer mechanic that challenges the normativity of code and digital technologies. Drucker says, “Algorithms are instructions for processes, for performances, whose outcomes may usually be predictable, but of course, are as open to error and random uncertainties in their execution as they are too uncertain outcomes in their use.”

The glitch defies planning, coherence, homogeneity, and perfect control, and most importantly, a glitch cannot come

---

15 Drucker, “Performative Materiality and Theoretical Approaches to Interface,” par. 11.
into being without the running of the program and the playing of the game. By playing through *Lim*, by working through its obvious, unobvious, and accidental rules, I have gained an understanding of the mechanics and mechanisms of the game without necessarily parsing the code itself. This understanding, this experience and knowledge-making can then become the ways and methods of playing, breaking, and (eventually) making other games. Without immediate access to the lines of code itself, I can still recognize, interpret, even feel, like an algorithmic kinesthetic sense, the *if/then/else* statements, the links between key and command, and the subtle (and not so subtle) shifts between subroutines for exploration, exposition, combat, cut scenes, success, failure, reward, even death. Kopas says:

Games can serve as sites for us to gesture towards queer utopias, to imagine alternative ways of being and living. For that to happen, we have to interrogate and rethink the work of playing. Mutating, breaking, and twisting games are valuable actions insofar as they help make visible our assumptions about play.¹⁶

All in all, it is this working, breaking, twisting, and imagining that underscores the myriad of ways that playing is making, a different kind of creating. Games are a constellation of textual, narrative, digital, mechanical, and cultural practices. Ask any player of digital games whether or not they are makers or builders and the response will be loud and affirmative. Playing and gaming makes all sorts of things beyond and in response to scholarly critique and theorization: high scores, fan fiction, save files, game art, machinima, strategy guides, bug reports, muscle memory, maps, notes, mods, reviews, communities, museums, schools, and not to forget heat, dust, radiation, noise, and tons and tons of e-waste. That said, I return to Ramsay’s invitation to expand and explore definitions of making and building. In this

---

case, the playing of games is one form of procedural literacy and performative materiality, which has

often been relegated to the domain of computer programming. [...] But the value of procedural literacy goes far beyond the realm of programming alone; indeed any activity that encourages active experimentation with basic building blocks in new combinations contributes to procedural literacy.¹⁷

Or as Tara McPherson argues,

In extending our critical methodologies, we must have at least a passing familiarity with code languages, operating systems, algorithmic thinking, and systems design. We need database literacies, algorithmic literacies, computational literacies, interface literacies.¹⁸

By playing digital games, I think we develop these “passing familiarities” but simply have not put names and numbers to what we take as “just a game.” Like my glitched square in Lim, it is possible to understand the inner workings of a system by looking at it from vantage points outside of it. Understanding code, understanding algorithms, understanding the ways digital technologies work need not be reduced to a Hamletian dilemma: to code or not to code. Rather, playing with code, experimenting with technologies can offer entrances and experiences that rely on an inclusive and everyday form of doing: playing a game.

¹⁷ Bogost, “Procedural Literacy,” 36.
Bibliography


