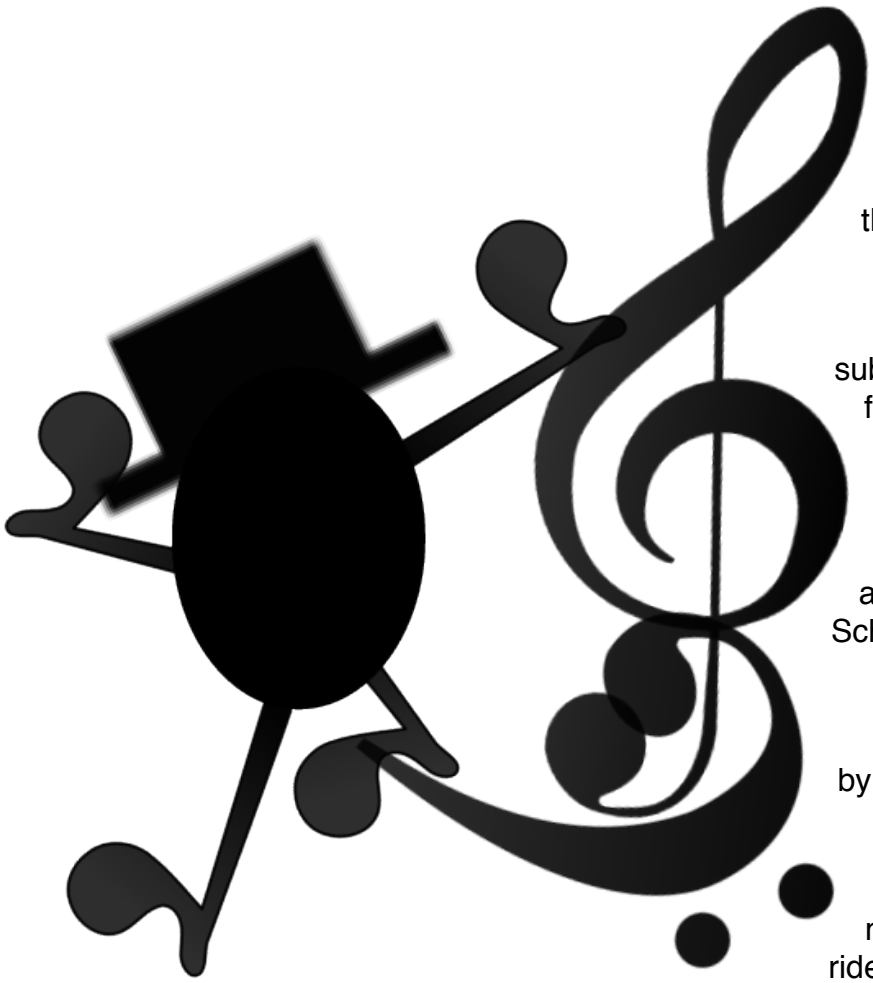
Rifftide



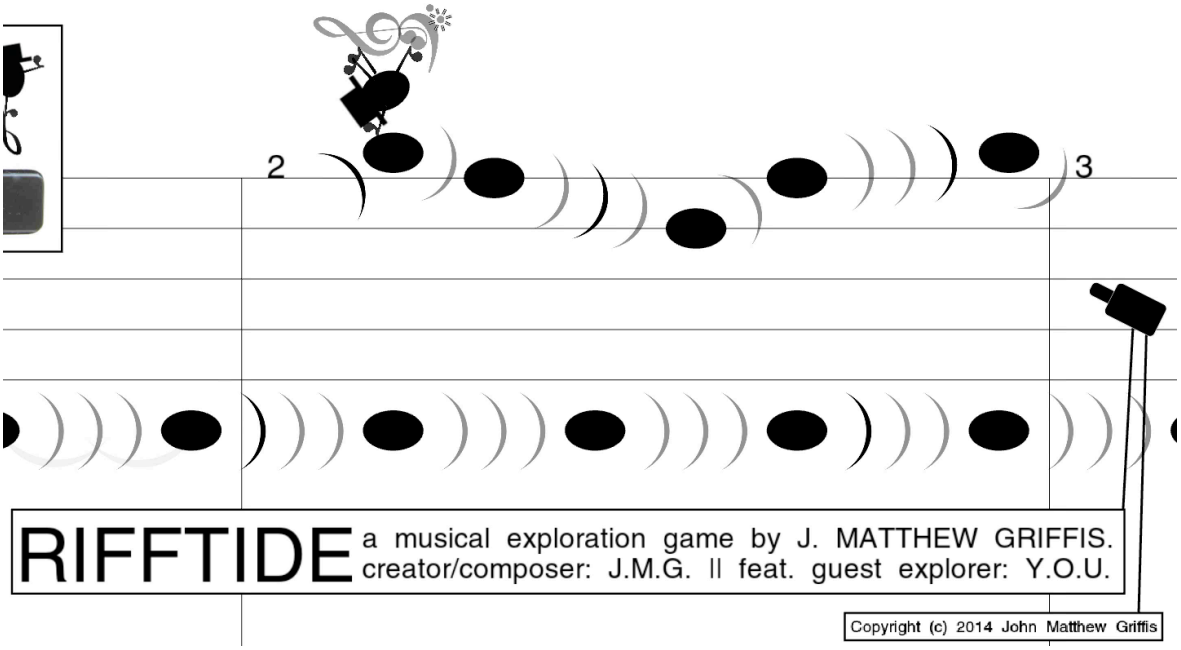
the game

the production document

submitted in partial
fulfillment of the
requirements
for the degree of Master
of Fine Arts in Design
and Technology
at Parsons The New
School for Design, 2014

by J. Matthew Griffis

music flows
ride the tide


2
3
RIFFTIDE
a musical exploration game by J. MATTHEW GRIFFIS.
creator/composer: J.M.G. II feat. guest explorer: Y.O.U.
Copyright (c) 2014 John Matthew Griffis

I / IV. Dedication

I dedicate this project to my fiancée, Trystan, and to my family, without whose support I would not be here, trying to see how many jokes I can fit into a masters paper about a bizarre music game. “'Tis wondrous strange.”[1]

I thank the many people at Parsons The New School for Design who helped bring this project to fruition, including (but not limited to) my thesis instructors John and Colleen, my writing instructor Barbara, and all those who played the game in its many incarnations and offered valuable feedback. I also thank those whose feedback was not valuable--they can’t all be winners.

[1] William Shakespeare, *Henry VI, part 3*, Act 2, Scene 1

I / II. Disclaimer

The following is a factual account of real events, told truthfully by those were authentically there, perhaps. Project names have been changed as needed to become more interesting. If dialogue has been edited, jokes sharpened or even birthed *ex post facto*, it is only for the sake of the amusement of the general public. Any inaccuracies or inconsistencies may be laid at the feet of narrative flow, and promptly ignored.

III / IV. Invocation

Sing to me, Muse,

of the game of many visions,

which was driven far afield,

after it had sacked music-rhythm's sacred citadel,

and quoth, "It's been done."

I. Abstract

Look at this (Fig. 1):

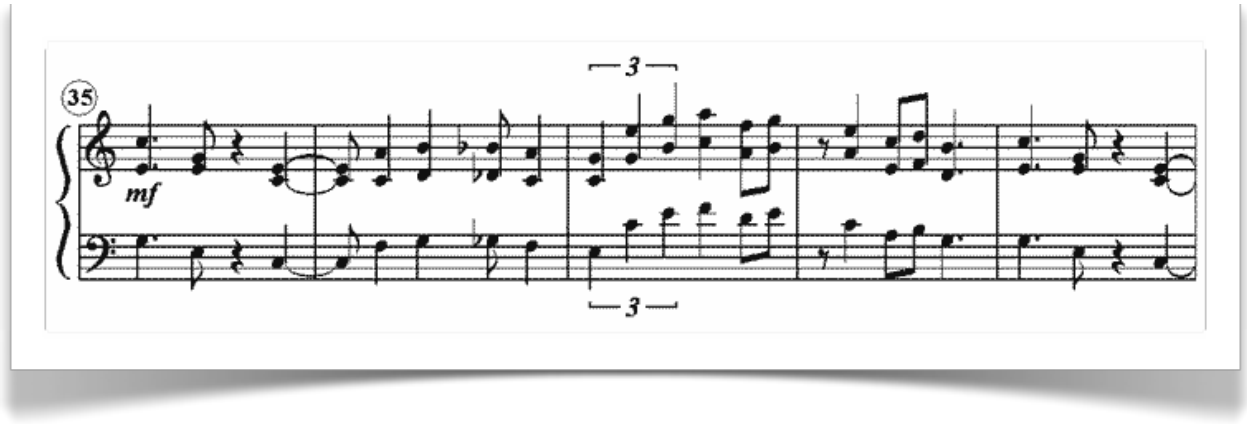


Fig. 1. ???

What is it? If you're like "duh, it's sheet music," two points for you, minus one point for sarcasm. If you didn't know what it is, not to worry--the others are only one point up.

People see radically different things when they look at sheet music. Some see the clear expression of an idea, reading it effortlessly. Others see a familiar symbology but struggle to interpret it. Still others see meaningless marks on a page. They are all right. Music is a language, to be spoken and heard, written and read, and as with any language, the spectrum of literacy is wide.

Music notation straddles the line between writing and drawing, description and graphic representation. It is spatial and temporal. Let us visualize our old friend that you thought

you'd left behind forever, the Cartesian coordinate system, and drop it atop the sheet music. The vertical axis (y) is space, while the horizontal axis (x) is time. The higher-pitched note appears above the lower. The note played sooner appears before the later. Whether you can "read" the notation or not, you can put your finger on the marks and trace their path--the flow of the music.

What if you could jump into that piece of paper like something out of *The Pagemaster* and follow the path yourself, jumping and climbing and sliding from note to note as if they were tangible objects in the world? And if you "played" notes by touching and interacting with them, what would happen? Why, you'd make your own music by moving through the space.

Rifftide is a video game about that.

II. Context

A. Domains

Tell me, self: where did such a strange idea come from? I'm glad I asked! I came to Parsons to study game design and made sure everyone knew it, so it would have blown minds if I didn't make a game for my thesis, and I didn't want to have to clean that up. But what sort of game?

While I always planned to incorporate music, it wasn't originally part of the gameplay. This being New York, capital of reckless experimentation, I had gotten into composition, and I thought it would be fun not only to create the game but the music, too. It was only after my instructor suggested that composing a handful of tracks would be a thesis-level commitment in and of itself that I thought "Pfft, what does he know?" Where was the faith? Nevertheless, I wondered if I could find a way to incorporate the musical explorations into the game design itself. Hang on...isn't there a genre for that?

B. Precedents

"I have an idea!" said designers. "Music is playful. Games are playful. Maybe we could combine them to create ~~useful work~~ maximum fun!"

This is neither a modern attitude nor limited to video games. For example, you might not know that Mozart was a game designer. His knowledge of computer programming may have been spotty but his knowledge of music programming made up for that shortcoming, and he and other classical musicians of the era created the "*Musikalisches*

Würfelspiel," which as we all know is German for "I wish I knew computer programming." These "musical dice games" consisted of a series of measures of music in the style of a minuet that could be combined in any order without sounding too much like a dying cat.[2]

Anyone could roll dice to compose their own version--the granddaddy of today's hot topic, the "procedural generation" that savvy video game designers use to create unique environments and experiences for every player. If the loss of authorial control enabled by the musical dice game's unprecedented freedom resulted in rather bland music, that was outweighed by the player's increased agency and sense of ownership: "It's *my* minuet, which has never been heard before!" For analogues in other media, see the work of the Oulipo, especially *Cent mille milliards de poemes* by Raymond Queneau.[3]

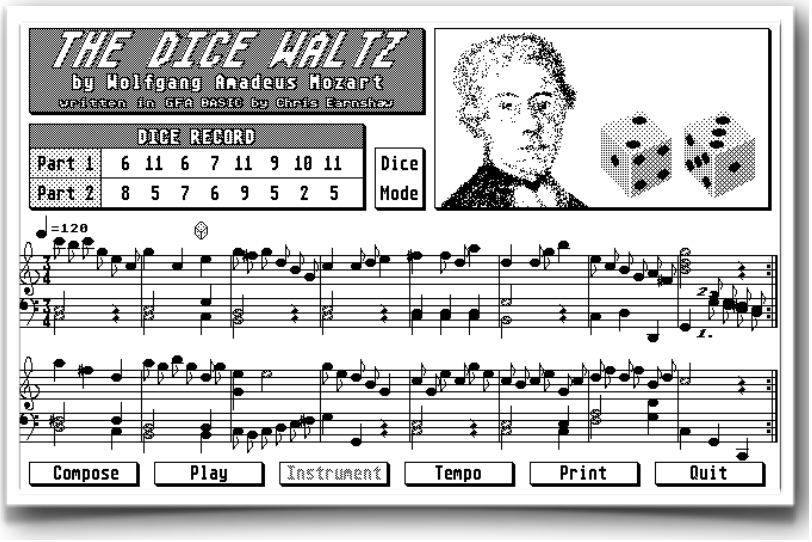


Fig. 2. A digital version of Mozart's game (www.amaranthpublishing.com/MozartDiceGame.htm).

---

[2] See http://sunsite.univie.ac.at/Mozart/dice/ and http://imslp.org/wiki/Musikalisches_W%C3%BCrfelspiel,_K.516f_(Mozart,_Wolfgang_Amadeus), accessed on 4/20/14.

[3] See http://www.growndodo.com/wordplay/oulipo/10%5E14sonnets.html, accessed on 4/20/14.

Ironically, Mozart's analog project may be spiritually closer to the final form of my thesis than any other precedent. Still, I fell in love with music-based video games long before I learned about the musical dice game, and they exerted a great influence on the project's direction. Many designers have used music as the basis for gameplay, with a variety of approaches. By far the most common is the rhythm game, which challenges the player to perform certain actions in time with popular music. It's a great way to experience music more actively, and it excels at creating the feeling of performance.

On that note, consider the king of music-rhythm, *Guitar Hero* (Fig. 3), which delivers the rock star fantasy *par excellence*. The odds are that you are, in fact, not a rock star, and will not now or in the future be standing onstage in front of a screaming crowd and performing the hits at life-threatening volume, but in *Guitar Hero* you can do these things. You "play" the songs by pressing specific buttons at the right moment in time with a scrolling fretboard on screen. While you can play *Guitar Hero* with a regular gamepad, this is like going to a costume party without a costume, and then trying to win the contest. Better to embrace your inner YouTube embarrassment and use the "guitar" controller, a piece of plastic shaped and designed like a reduced-

Fig. 3. Guitar Hero III *(PlayStation 2)*

scale guitar. You have to press buttons on the neck of the guitar with one hand and "strum" a button on the body with the other, much as you would a real guitar. Between the authentic feeling of the movement and the way the note patterns onscreen approximate the actual progression of the song, it does feel like you really are a) making the music and b) a guitar hero--no small accomplishment.

And now for something completely different (but also rhythm-based): it's *Elite Beat Agents* (Fig. 4), a superlatively bizarre game. Take control of a group of male cheerleaders (modeled after the Japanese *Ōendan*) who turn up like firefighters in tuxedoes wherever there's a problem, then (the analogy breaks down here) dance the problem into oblivion. The role of the player is to help them do this by tapping, dragging and scratching on the DS's touchscreen in rhythm with the music. Very similar to Guitar Hero, except that in this case there's no illusion of actually playing the music, no deep physical mapping of the interactions onto the musical progression (sadly the game does not

Fig. 4. Elite Beat Agents *(Nintendo DS)*

include a suit controller). It's pure rhythm. Taps match to percussive beats or other identifiable "moments" in the song, and so on with the other interactions, but that's as deep as it goes. As with Guitar Hero, the music is there to be enjoyed and to make it easier to play the game, but you can play EBA with the sound off, too.

Then there are games that turn away from the "you are performing this song" feeling of music-rhythm toward a sort of synaesthetic experience, in which gameplay and music are so tightly bound that the line between them blurs.

A sublime example is *Soundodger* (Fig. 5). You can play this trip of an experience for free in your browser right now, and I encourage you to do so instead of reading the description that I am about to attempt. In *Soundodger*, the player chooses a music track, then moves a tiny circle about the center of a circular arena, trying to survive the duration of the music while circles around the periphery fire a storm of triangular attacks inward. At any time the player can slow down time to make it easier to dodge. What happens in the music manifests in the game--for example, certain enemies are the "drums" and attack in sync with the track's percussion. The player does not generate the music or interact with it directly; rather the

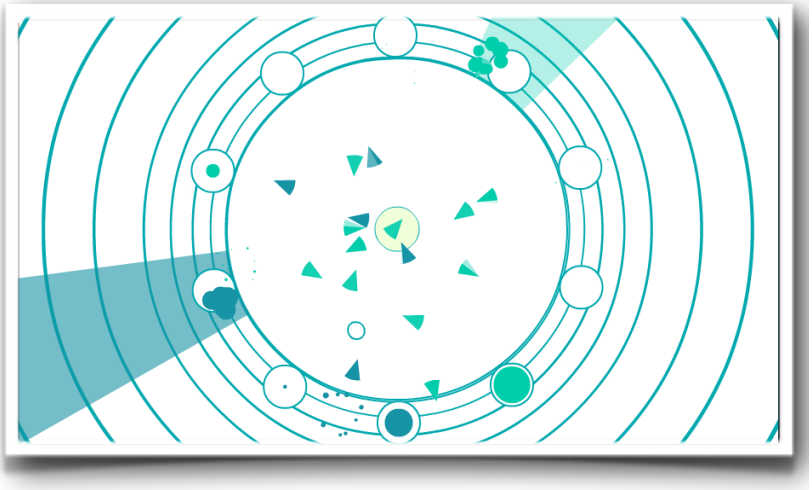


Fig. 5. Soundodger *(browser)*

game's world acts as a music visualizer and interpreter. Players who listen closely to the music can anticipate what's coming.

The previous two categories turn a fixed music track into gameplay, one by expressing it through the player's actions, the other, through the game's actions. What about the games that enable the player to *change* the music, to create an original sound? There are only so many times you can play through the same music you've heard on the radio a million times before you start thinking about whether life has any meaning and need to create something to combat existential crisis.

Such games do exist, but they are hard to find. In these games the player exerts greater control over the soundscape than the simple binary of "music on / music off," by generating sound directly or indirectly through other actions. Two examples follow.

Fig. 6. Synthpond *(iOS)*

Synthpond (Fig. 6) is a really fun game. Sure, it doesn't have "goals" in the conventional sense... or challenge... or levels... OK, it's a tool. This category is

so barren though that better examples are hard to find, and *Synthpond*[4] turns composition into a playful and spatialized activity in a unique and relevant way. Imagine dropping a rock into a pond and the ripples that it makes are sound waves, and when the sound waves hit the waterlilies they resonate and send out their own sound waves, and all the frogs stay away. It's like that, only more abstract. The player drops nodes into the pond, setting their pitch and whether they will generate or receive waves, and because this is a 2D space and the waves move at a certain rate and the nodes can be placed anywhere and set up any way, the pleasure is in experimenting with what kind of soundscapes one can produce.

Fig. 7. Beatbuddy *(Steam)*

Then there is *Beatbuddy* (Fig. 7), an interesting counterpoint to everything discussed so far (but definitely a game). In *Beatbuddy*, the player remixes the soundtrack, live, through gameplay. The mechanics themselves are traditional (move through a 2D world and overcome obstacles), but every component of the soundtrack is attached to creatures in the game, with which the player can interact, turning on and off the associated audio at will. Say a clam holds the percussion, and

4 Created by Parsons MFADT alum Zach Gage.

you attack the clam so it retreats into its shell. No beats for you! At least until you can convince the clam to open up again.

“Hang on,” you should say if you’ve been paying attention. “Isn’t this the very same simple, binary on/off system you wanted to escape when discussing the music-rhythm crowd?” Yes, but given the fact that it applies to individual components of the soundtrack (rather than the whole thing), as well as the player’s ability to change the syncopation of the various components to one another, it’s possible to change the music as a whole and to create a unique sonic experience.

C. The categories: pros and cons

While there are many other precedents I could cite, and many approaches varying from those described above, as a general rule they may be separated into the following three categories, each with their own strengths and weaknesses.

The first category I call “**mimicry**” because, though the details of the implementation vary, the play is in mimicking a musical performance, through some kind of abstracted interaction that maps in a physical way onto the music. Everything about the performance is fixed. There is no way to change the music itself--diverging from the prescribed rhythm and actions results in failure. Embrace the tyranny of musical perfection!

In this group I include the music-rhythm games, like Guitar Hero, as well the rarer (but very popular) pitch-based games about singing, like *Karaoke Revolution*. All of these excel at simulating the feeling of performing popular music, but the player only has real control over how accurately she follows the Simon-Says-like instructions. Consequently, there's no real musicality and no opportunity for creative expression.

The second category, **synaesthesia**, is similar to mimicry in that listening to the music makes the gameplay easier. In this case though the actions of gameplay are not direct representations of performing music. While the music and gameplay work together very intimately as in the example of *Soundodger*--which is, after all, why I call the category "synaesthesia"--they exist and may be conceived of separately. Turning off the sound in *Guitar Hero* defeats the whole point, because the actions lose their significance--you're pressing buttons, but to what end?[5] Turning off the sound in *Soundodger* certainly diminishes the richness of the experience but preserves the meaning of the survival action. Dodging is dodging. Ah, ha, ha, ha, stayin' alive.

Thus the connection in this category between music and gameplay is more profound, less about causation (pressing a button causes a sound to play) and more about transcendent expression (when a sound plays, something glows). This is fascinating territory and well worth exploring, but again it's not really about gameplay centered in musicality, creative or otherwise. Indeed, some of the most synaesthetic games would likely not be labeled "music" games at all (*Super Hexagon* springs to mind).

---

[5] A question for the technological ages.

The third category does promote actual musical creativity, experimentation and **composition**. It sure is difficult to do, though. *Beatbuddy*'s gameplay itself is pretty traditional, it just has an additional sonic layer. *Synthpond* is more of a toy than a game. There's a huge knowledge barrier to overcome to turn composition into gameplay. No doubt the reason music-rhythm is the dominant genre is because it's so accessible--most people have some sense of rhythm, which cannot be said for musical aptitude or interest in the bogeyman known as "theory."

Furthermore, rhythm-driven or synaesthetic gameplay generally doesn't ask much technically of the player, at least not as far as the music goes. Something that embraces other aspects of music and demands real musicality is hard-pressed not to exclude many people. This is likely why there are so few games in this category. Even *Beatbuddy* limits the player to the music tracks included in the levels, however they may be remixed. The music depends on the gameplay, but not vice-versa.

There is so much richness in music beyond rhythm that it seems a shame not to explore ways of expressing all that creamy goodness through gameplay, despite the inherent challenges and likelihood of alienating people. It falls to me, then. Let's do this thing.

III. The Prototypes

Given that most of the precedents served as examples of what not to do, but that I thought *Synthpond* and *Beatbuddy* were on the right track toward something new, it would only make sense to slam them together repeatedly at high speeds until they became a thesis project. Thus the original idea might best be described as merging *Beatbuddy*'s approach of familiar gameplay mechanics and tropes with *Synthpond*'s transformation of musical notation into a physical space.

That analogy came later, though. In the beginning, while thinking about what other parts of music to express through gameplay, I hit upon the idea of teaching rudimentary music theory, surreptitiously so the player wouldn't realize education was occurring and set the computer on fire. My inspiration came when I contemplated the spatial nature of sheet music, and then put some next to a screenshot of a *Super Mario Bros.* game for some reason (Fig. 8).

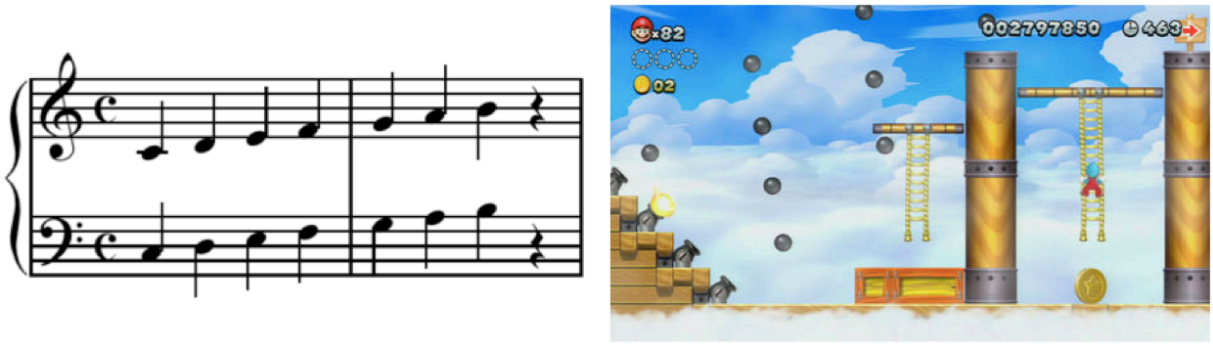


Fig. 8. Sheet music and Super Mario Bros.*--separated at birth?*

Those cannonballs flying through Mario's sky...draw some staff lines behind them and they could just as well be musical notes. Why not invert that idea by turning a piece of sheet music into a world to be explored, mastered, and--carefully, very carefully--even learned?

The thing about doing something new is that there's no old to provide guidance. "The world is sheet music!" is a good start, but doesn't say much about what that actually means. Consequently my vision for this project transformed more than a store full of action figures. I discuss the effects both practical and psychological of that in the "Challenges" section (hint: it was challenging). As you can probably guess though, it made for a lot of prototypes.

A. Killer Riff

The first real prototype (which benefited from a semester's worth of ideation and development) used a record-and-playback mechanic that deserves more exploration, which I do say so myself. The premise went like this: it's a platform game like Mario where you run and jump through a 2D environment, but the environment is the musical staff and some of the

Fig. 9. [First prototype] Who needs the ground when you can make your own platforms?

platforms are musical notes that sound their tones when you get close, and you can record a note and then play it back elsewhere to make a platform in the new location (Fig. 9), and you have to do this strategically to get past obstacles and make it to the end.

It's probably worth noting that just prior to developing this prototype I had played through *Braid*, a living painting of a game where you manipulate time to solve all kinds of environmental-manipulation puzzles and think difficult thoughts about relationships and being human, after which I thought "*Braid* is cool. I should ~~make Braid~~ make something like *Braid*." Indeed, with no hazards and no time pressure, the tutorial I built for the prototype was a deliberative experience inspired by *Braid* as well as the decidedly not-deliberative *Kirby Canvas Curse* (where you draw platforms on the touchscreen to keep Kirby going).

The prototype had a second part, the boss battle captured so dramatically in Figure 10. In this more action-driven scenario a boss enemy would attack the player with a certain riff, i.e. a sequence of notes forming a coherent musical idea. The player would need to record the entire sequence, then play it back in the proper order to defeat the boss. Hence the

Fig. 10. [First prototype] A very intense boss battle.

project's original title, "Killer Riff." Really things could only go downhill from there.

Playtesting the prototype provided an excellent lesson in the importance of clear instructions. It turns out that if part of your puzzle design is to never explain anything to the player, you're doing it wrong. I did what I could to improve the feedback, but resisted providing any guidance as overt as "press this button to record a note!" out of a pure-hearted conviction that the design would speak for itself. Inevitably people struggled in the face of my self-righteous opacity until I couldn't bite my tongue any longer and stepped in to explain things. Once I did, the game was a hit! Ship it and include one Explanatory Matt in every package.

At the end of the semester, I presented this prototype to guest critics for feedback. They were more interested in the dynamism and musicality of the boss battle (concept, if not execution) than in the slower-paced puzzle-solving and the disjointed tones of the tutorial, and they questioned the feasibility of developing compelling puzzles around the recording/playback mechanic. They encouraged me to push the musical side of the project more aggressively.

Alas! Poor boss battle. I never did finish implementing you.

B. Dancing Fingers

I really liked the first prototype and its vision. Look for that to make a return in a future project. But I struggled with how to deepen the experience to be more than a prototype,

and I agreed with the critics about emphasizing musicality. Consequently, after a very busy winter break of not working on my thesis, I decided to prototype a radically different gameplay mechanic.

I play piano, not with any especial level of skill, but one of the things I like best about the experience is that occasionally I enter a flow state, in which I lose my sense of self and just groove to the music. I imagine this is what people experience while dancing, but since I am much too self-conscious to achieve the requisite level of abandon, I tickle the (metaphorical) ivories instead. In the second prototype, inspired by this experience, the player controlled two onscreen hands simultaneously, steering linear passage through a set musical sequence and striking notes with the hands to sound them (Fig. 11).

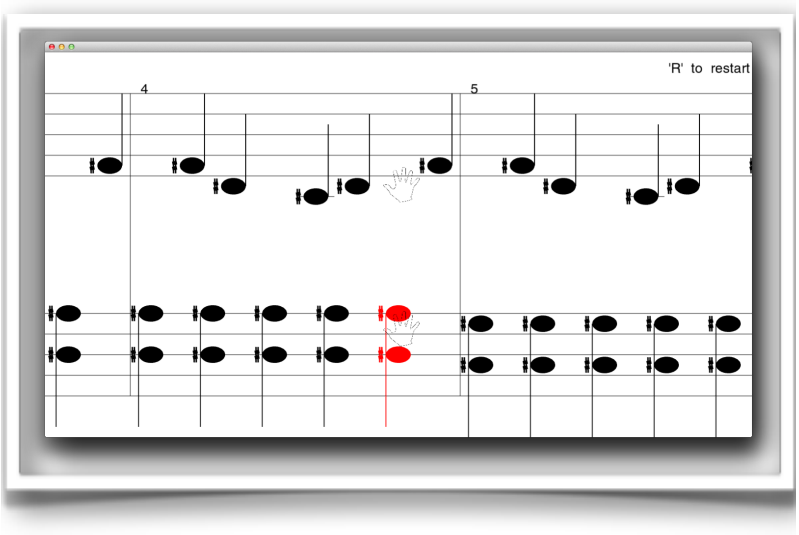


Fig. 11. [Second prototype] Steer the hands to play the groove.

Too literal, do you think? It certainly nailed the increased musicality. In the previous prototype the avatar could move forward and backward at will, and notes had to be positioned to work as platforms, both of which affected how the experience would

sound. With no such constraints here it was easy to design (i.e. compose) a playing field that sounded good.

Furthermore, playing the game was sort of like “reading” the sheet music, in that by steering the hands to follow the notes, the player followed the flow of the music. And (this was my favorite part), since the notes only sounded when struck, it was not only possible but downright easy to experiment with a sort of on-the-fly remixing, choosing which notes to hit. Finally, since this prototype embraced the full two-clef musical score notation, it better communicated the “sheet music world” context.

That all sounds great--ship it! Wait, retrieve it! There were cons to this approach too. The composition/remixing as an interaction was limited to choosing not to hit certain notes, not exactly an advance over *Beatbuddy*. If the goal was to create a flow state, the precision demanded by a “hit all the notes to stay alive!” system belied that, yet there wasn’t another gameplay framework to take its place. And the experience was too simple, with little challenge other than the mental gymnastics of coordinating both hands. One playtester said it felt like “a Flash game about how to play piano”--not a compliment. This mechanic had gotten away from the idea of the notes as tangible objects in space. The next prototype would be an attempt to restore that.

First, though, I decided to combine all the prototypes into one and build a beautiful menu system so I could switch between them easily (Fig. 12).

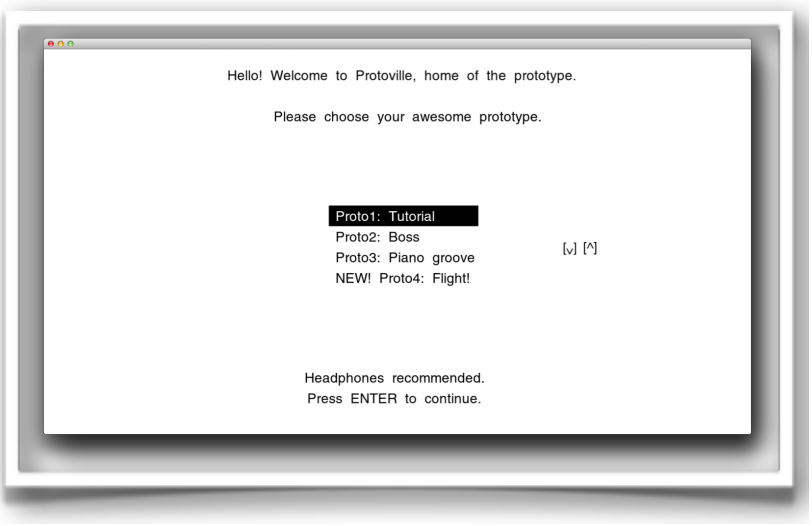


Fig. 12. The exciting new menu for prototype selection.

I also added controller support, so people could play using an Xbox 360 gamepad. Surely the thesis must be done now, right?

C. Flappy Note

It was at this point that our thesis instructor gave a rousing speech in which she urged us to stop playing it safe and really go crazy with our project ideas, then dial it back as needed. Days later, I attended the video game convention IndieCade East and sat in on a presentation about music and games, where I first learned about Mozart's dice game. I got inspired again and created another prototype (this might be an allergic reaction).

This third prototype (Fig. 13) was a variation on the previous one. I restored the action to a single avatar and put the fan-favorite rectangle back instead of the hand, but this time it

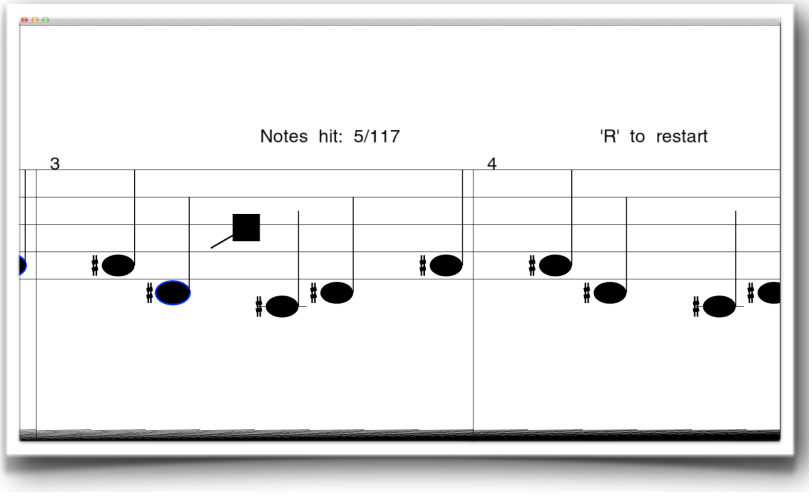


Fig. 13. [Third prototype] Flap your way to sweet music.

could flap upward or dash downward (while continuing to move forward automatically) and had to hit as many notes as possible while trying not to plummet to Earth under the cruel sway of gravity.

I tested this new prototype extensively and discovered the controls had major issues. Still, players' feedback was mostly positive, and many tried the new prototype over and over again to go for a higher score. One said she liked that the better she played, the better the music sounded, but that even if she didn't play "well" the music still sounded good, and that she could even choose which notes to hit to create her own soundtrack--very gratifying to hear, since this was my intention!

Most of the testers were new to my thesis, so I took advantage of the shiny new prototype-selection menu to show them the earlier material too and solicited feedback on how they felt about each prototype. They felt just fine, thank you very much, and multiple people recommended combining the various types of gameplay in the final product. This too was my intention. Call the Nobel committee and ask them when they're going to start awarding a prize for games, because my project is a lock.

What with all this back-slapping I was in danger of needing medical attention, so my instructor stepped in to apply the stinging balm of a reality check. I should be much more concerned about the control problems, and more importantly the one-dimensionality of the play experience. So I revealed to her my secret weapon: a wild and crazy vision for the entire game, incorporating narrative and other mechanics not

yet prototyped. What could go wrong? She urged me to make a strong presentation to the guest critics coming in the following week, describing the whole project and its context, without worrying about whether I'd yet built the systems.

D. Interlude: the Bassosaurus' roar is the cue for my solo!

With that encouragement, I stepped up my presentational game and delivered to the guest critics mockups, a new project title, and the following elevator pitch:

"Jump, surf and fly through a musical-score playground in 'The Mighty Bassosaurus.' Build and change the levels as you play them, and create your own soundtrack with your every move, in this action-music game. No musical knowledge required."

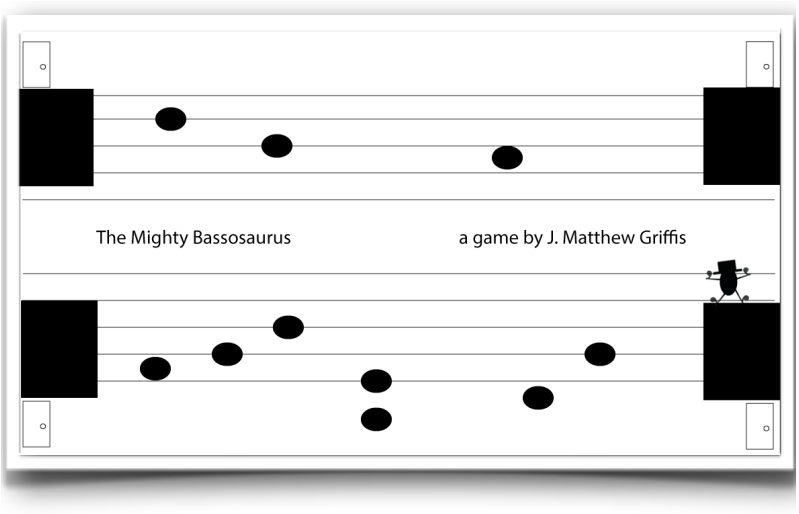


Fig. 14. [Mockup] The hub world, where the player moves between levels.

Pretty sweet, eh? I'd give the Nobel prize to that game. As the action verbs in the pitch's first sentence imply, the gameplay would have three components, illustrated by the aforementioned mockups. First up, the "hub" world, where the player would move between levels (Fig. 14).

What's that thing with the hat in the lower-right? It's the new avatar, of course! The old standby Generic Square is replaced by a jolly looking little fellow composed of music notation. The hub would grant the player initial access only to the first level and require the use of the platforming and recording/playback mechanics of the Killer Riff prototype to reach the other levels.

The second type of gameplay would be a replacement for the Flappy Note prototype, dubbed "Surfing the notestream" (Fig. 15).

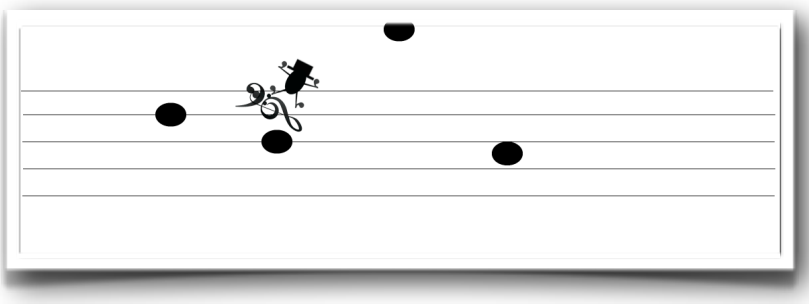


Fig. 15. [Mockup] Surfing the notestream in the second type of gameplay. Not pictured: the actual stream.

I decided to move away from the flapping mechanic of the earlier prototype because it didn't make sense within the rest of the game's context. If the notes were physical objects that obstructed the avatar's movement, how come the same avatar could blithely flap right through the notes while in flight? Replacing this with a surf/skate/snowboarding-style mechanic in which the avatar would ride atop a stream of notes, following their course--inspired by my instructor's comment that she'd like to see the notes connected, indicating the path she should follow--preserves the "follow the flow of the music" idea while remaining true to the game's internal logic. Also, surfing is cool.

The third type of gameplay would be the most experimental of all, a “soloing” mode inspired by the Dancing Fingers prototype in which the player could fly freely up and down while moving always to the right and (diverging here from the source material) could create new notes at the press of a button.

Figure 16 shows not only the mocked-up gameplay but also an initial take on the “Mighty Bassosaurus” of the project’s title. This fearsome creature would pursue the player like the T-Rex from *Jurassic Park*. Only by creating notes strategically (that is, composing, but don’t tell anyone) could the player survive--in this case by leading the note-hungry Bassosaurus into an obstacle that would slow it down. Meanwhile there would be a musical accompaniment playing offscreen so that the player would have something to play against, in both senses of the word.

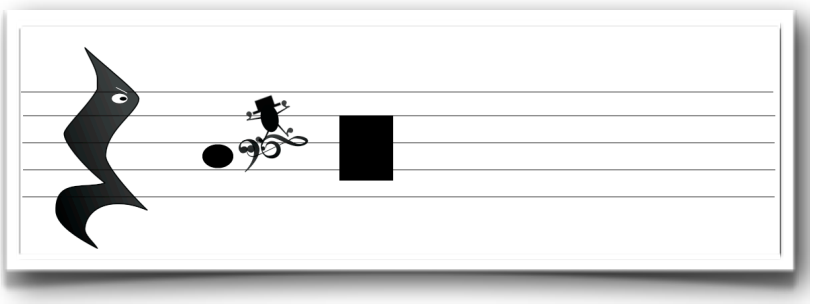


Fig. 16. [Mockup] Making notes in the third type of gameplay. The Bassosaurus is in pursuit!

The game would alternate between these three types of play, as the player completed the various levels, culminating in a final battle against the Bassosaurus in which the player (this is where it gets really juicy) would have to recreate in soloing mode a specific musical riff in order to defeat the Bassosaurus and win the game. Sound familiar? Yes, Killer Riff’s boss battle makes its return! That winning riff would be hidden

somewhere in the game's world for observant players to find and other players to look up on the Internet.

The presentation was mostly mockups but I did show a new prototype of the composition mechanic, *sans* Bassosaurus sadly (Fig. 17).

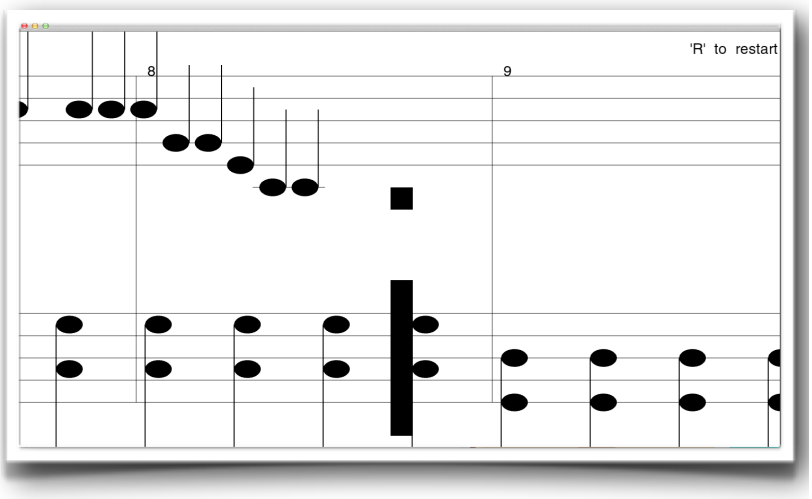


Fig. 17. [Fourth prototype] Composing! The bottom is on autopilot while the player composes in the top half.

Feedback from the critics was generally positive, with kind words for the ideas and mechanics as well as the crude but charming art style of the mockups. They were concerned about the absence of clear gameplay goals and wondered what would bind the three disparate gameplay mechanics together. One critic suggested that the surfing mechanic could offer a beautiful new experience of music as terrain, but that getting the feeling right was key. What's that sound? It's the sound of "time for a new prototype."

E. Surfin' USofA (not trademarked)

I remember it well: the rush of enthusiasm following the presentation; the burst of energy as we heading immediately into Spring Break (more of a cruel joke than a reprieve for Thesis students); the confidence that I could "prototype the surfing mechanic over the first weekend" then spend the rest of the so-called "break" building

out the experience of the mockups; before it was all swept away in the deluge of the notestream.

In reality, prototyping the surfing mechanic proved to be extremely complicated and time-consuming. Who would have guessed? This may have had as much to do with a flawed prototyping methodology as it did with the scale of my ambition, because I spent a lot of time and got very intimate with trigonometry while trying to make the stream connecting the notes dynamic and animated when in fact crude lines would have served the immediate purpose, but whatcha gonna do? The project was due soon. Stuff, including aesthetics, had to get done.

Adding in the character from the mockups was easy enough, and in a stroke of genius or perhaps madness I built a physics system for his hat so it would soar into the air given sufficient momentum. This went a long way toward adding some much-needed personality to the experience--everyone loves hats! And if the hat sits at a jaunty angle and you can “pop” it like a yo-yo, well, ship it, because the game is pretty much done.

I could have flown to Majorca at this point, but I needed something to do while I waited for the hat-fueled streams of money to come pouring in. I decided to keep working on the project and maybe add a second hat to sit on top of the first one. Maybe this could be like that Dr. Seuss story with all the hats![6] In fact, maybe I should abandon the music motif altogether and just do a hat simulator.

---

[6] The 500 Hats of Bartholomew Cubbins

In spite of all the hurdles, I did finally get the surfing mechanic working (Fig. 18), and when I playtested it, the hat was a hit! There were issues with the surfing. The way it worked was that the player could rotate the board clockwise or counter-clockwise, as though shifting weight toward the front or back, and if riding in the stream at the time, rotating the board out of alignment with the stream would slow everything down to quarter-speed. In theory, this was a good idea--keep your weight and board aligned to the stream to stay at full speed and not fail horribly, just like real boarding! In practice, the slowdown sucked the life out of the experience and there wasn't even enough feedback to explain the desiccated husk.

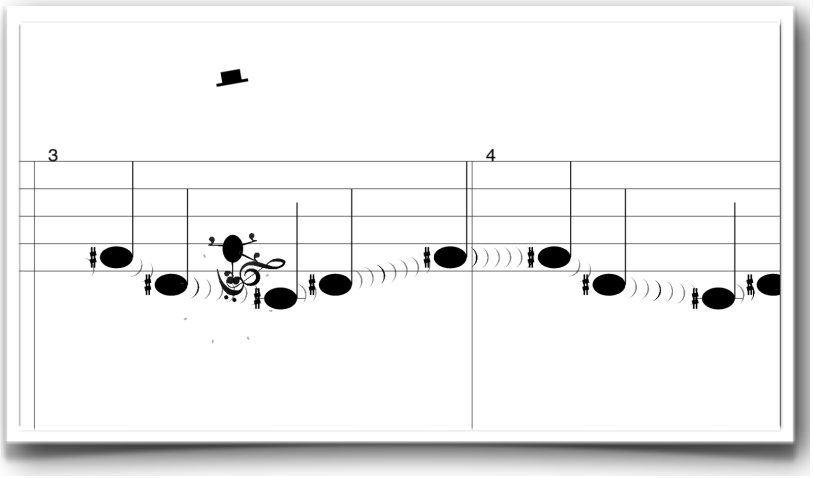


Fig. 18. [Fifth prototype] Toss your cares and ride the stream.

I decided the slowdown was too binary to work well--if the only options are right or wrong, full speed or painfully slow, there's no room for nuanced play. So I rethought the effects of the rotation. Now it would not affect the speed, but if you tilted too far in one direction or the other, you'd fall over and fail the level. Furthermore I added a kind of positive reinforcement, so that if you leaned far enough, the stream itself would start pushing you further in that direction and you'd have to counter the force. Incentive to align to the stream? Yep. Heavy-handed? Nope.

Playtesting revealed another big issue, which was a lack of meaningful choice. The player didn't have much to do beyond jumping and trying not to slow down or fall over, which is enough for the playground and intoxication tests but not for a music game that's supposed to be about creative expression. When testing the Flappy note prototype my instructor had suggested I include multiple paths and let the player choose which to follow. I liked this idea because it would be a clever middle ground between the unfettered and paralyzing freedom of composition in a void and the restrictive perfection of prewritten music. This would be my musical dice game, allowing me to control the components to ensure the result would sound good while allowing the player some creative ownership and a sense of individualized experience.

I created a surfing tutorial and built additional paths into my existing level, then put the improved experience to the test. The tutorial was pretty dull, but the overall player feedback was more positive, enough to convince me that I was on the right track. What was needed now was better in-game feedback (to make the player's actions feel significant), more content, and possibly additional mechanics and goals--that is, an actual game. And more hat. People loved the way the hat flew about and enough people asked for it to be part of the gameplay in some way for me to take the idea seriously.

One other minor concern bears mentioning at this point, which is that the project due date was just weeks away. And not the good kind of "weeks," like 12 weeks or 52

weeks, but the “how many weeks is a month” sort, the “keeps you up at night” variety. Given this alarming time pressure and the work that remained to be done to get the surfing gameplay to a sufficient level of quality, it didn’t seem realistic to attempt to implement the full scope of “The Mighty Bassosaurus.”

Consequently, I dedicated the remaining time to fleshing out the surfing gameplay, polishing it to shine like the sun over California waves, and exploring its potential as the foundation of a complete and coherent experience. The result, the culmination of this musical odyssey, is Rifftide.

IV. Challenges

This might surprise you, but the road between thesis concept and execution was not smooth! There were many challenges along the way, and here they are to supply your daily dose of *schadenfreude*, starting with the first and most important point:

A. Going it solo--was I crazy?

Yes.

B. Finding the focus

When I presented the concept for The Mighty Bassosaurus, I laid out the...hang on, I suppose I should say a little more about that previous point.

A. Going it solo--was I crazy?

I did not plan on doing my thesis alone. Most of the thesis games from previous years that I learned about were group projects, and there were plenty of people singing the praises (and practical necessities) of collaborative game development. Making a video game real isn't just designing it, but also programming it, producing the art for it, producing the sound for it, testing it, processing the feedback from testing it, redesigning it, reprogramming it, redoing the art and sounds for it, retesting it, and so on *ad infinitum*, or rather until the deadline. It's a lot of work, which isn't to say "oh, poor me having a hard time making my first-world entertainment experiences" but rather to emphasize the value of having one or more people to share the responsibilities, to complement one's own skill set, and to provide focus (both points I will examine further in following sections).

So why didn't I recruit another student to work on my thesis? You can probably anticipate the difficulty. I tried, half-heartedly, to find someone, but any such individual would have to subsume his or her interests to mine, because it was *my* thesis idea, not something shared. Considered in this light, it's amazing that collaborative projects happen at all, because unlike in the games industry where people work for other people, for a collaborative thesis to work both students have to be animated by the same passion.

As such, I had the following options:

1. *Give up my idea and come up with another in concert with someone else*--this was problematic because I really cared about my idea and wanted to make it! And there was no guarantee that the prospective partners would be any less enthused about their own ideas, so either they'd have to compromise too or else I'd have to work on their idea. Not much of a solution.

2. *Coerce another student into working for me*--people are always falling asleep in compromising positions all over campus; perhaps I could threaten to post the pictures on Facebook and embarrass them unless they helped with my thesis. This wouldn't work because people already do that and it doesn't change anything except how much people laugh at Facebook. The call of the nap is wild and irresistible.

3. *Pay someone to help me*--I could hire outside help. This is actually something people do and would probably have been a good idea for specific tasks (draw these graphics!), but would require something that was in very short supply throughout the process, which is clarity of purpose. It would also cost money, which is a sticking point, and anyway what would I outsource? I liked doing the design and the programming, and described my infernal determination to make the music too. This also wasn't practical.

4. *Go solo*--what else is left? For a passion project with an experimental and fluid bent, the only practical strategy was to do it all myself, come hell or high notestream.

B. Finding the focus

When I presented the concept for The Mighty Bassosaurus, I laid out the project's core goals (Fig. 19). It was nice to realize that these goals were consistent across all the many prototypes--a clear sign that the same fundamental vision drove the process, even if they were never previously articulated so clearly.

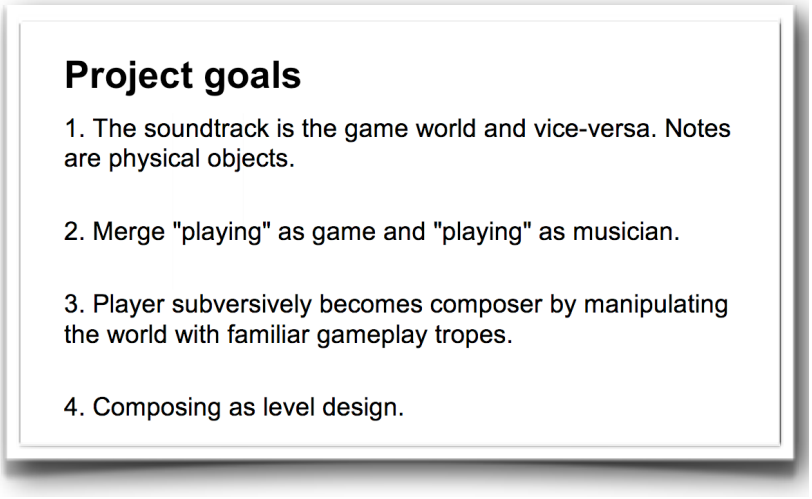


Fig. 19. The project goals--surprisingly consistent throughout (though not articulated until late in the process).

That was about the only consistency. I really flailed about while trying to figure out how to implement this exciting but vague concept, which is why the prototypes took so many forms. Looking at them now, I can see how they all built upon one another, that they are not completely disparate. Riffide is indeed the heir apparent of what came before. But my reluctance to commit to a specific and narrow vision, my insistence on experimenting over and over again with new mechanics (which often proved time-consuming to implement) without a clear idea of what I wanted, really cost me the opportunity for mature development of any implementation. Rifftide, whatever its merits, was intended as only one component of a larger game experience, but I ran out of time.

C. Technical challenges

Programming! It's fun but hard. I coined the phrase "the agony and the ecstasy of coding," which I'm sure no one has thought of before. It's accurate, though. A lot of programming is like banging your head against a wall for hours, and then when you finally break through the wall the excitement makes you forget how much your head hurts.

Too grim a metaphor? That was probably an excessive Freudian slip, but in actuality the experience is that of struggling with a difficult problem, and really getting frustrated because of how much time it feels like you're wasting in pursuit of a solution you may never find, but then when you do find it, and something amazing happens and YOU DID IT!, it's a wonderful feeling. Order out of chaos, beauty out of raw materials.

Puzzle fans know this. Coding is logic puzzle-solving. If you like logic puzzles check your back because you may already be a coder. The computer will do whatever you tell it to; the thing is that it is very stupid and literal-minded, so nuance, subtlety and "you know what I mean" is lost on it. It does not know what you mean. If you say "computer, kill all humans," it will do its best to kill all humans, not even slightly realizing that you meant it ironically in the spirit of anxious Cold War-era science fiction. The computer only knows what you say, exactly as you say it, so if you slur your words or forget the correct punctuation or (heaven help you) get your own logic mixed up, well, the result will be a head-banging session. And there were many.

Then there are the technological failures. One time recently all the audio files for the project disappeared from my computer. No explanation, no theater, and certainly no logic. One moment they were there, the next, my game was silent as the breath of the Reaper. I can only assume they'd had enough of listening to me pretend to have mastery of trigonometry while I painfully drew out angles on scratch paper yet again. Fair enough. Still, it left me in a bind, especially since I had rather recklessly not backed them up alongside the rest of the project. Hang on, I could retrieve them from one of my earlier prototypes! Oh, they'd disappeared from those (completely separate file folders) too. Isn't technology grand?

It is, because I had another automated backup that is a lot smarter than I am, and I found the missing files there. I restored them and decided they could join the official project backup party. There was some grumbling in the ranks about being stuck back in the project when they had only just escaped, but I promised the wayward mp3s that I would layoff the trig for a while, and they seemed content.

D. Who needs graphic design?

I do not have a graphic design background and visual design is my Kryptonite, if you'll forgive me comparing myself to Superman. I'm learning things, slowly, and developing some sense of style, and I took some drawing lessons back in the day that I would love to revive, but as a general rule, I prioritize visual polish last. This is an issue for a project that needs to attract eyeballs, especially one that would be presented in a gallery setting. My fiancée is good at drawing and I approached her about making the game's

art (free labor!). But that would require the clarity of vision (in this case concerning how the game should look) that was so lacking.

Fortunately, I found a workaround, which is music notation itself (free graphics!). Given the game's setting, it made sense to use the visual elements of notation for every visual element of the world, which has turned out really well in the case of the avatar, his half-rest hat, and the music-clef surfboard.

Still, that only gets me so far. A game needs visual style beyond the simple graphics of the in-game objects, which I've had to supply myself. It's unlikely the game will impress visually (though animation and feedback effects carry some of that load), but embracing the minimalist, notation-based aesthetic so that things at least look consistent will hopefully be sufficient to convince people to try the game and discover what fun there is to be had.

E. Playtesting

This goes back to the difficulties I had with finding the project's focus. Because I was constantly making major changes to the game design, and because I took so long to settle on any one direction, I playtested rarely, either because I was busy actually coding the new mechanic that needed testing, or because I hadn't had time to make enough changes since the last playtest to make a new session useful. This shouldn't come as a surprise to anyone who creates things, but it's also hard to show something

you made to other people and expose yourself to the possibility that they won't like it. It's something one just has to get over, but that doesn't mean I'm very good at it.

I should note that I did still do a lot of testing, and got a tremendous amount of useful feedback out of it, which is why I thanked people for it at the beginning of this paper. But there's no doubt that, project permitting, more testing would have done much to improve the game and also spread the word about its existence (itself a challenge). When I did test, I would often show people all the prototypes in the hopes of finding out what was "better," which is another thing that makes sense in theory but in practice likely diluted the feedback I received on any individual prototype. The playtesting lacked focus as much as the project.

V. “Post-mortem”

I don’t like the expression “post-mortem” for this kind of thing because I don’t think my project is dead. Rifftide lives! Music is life! But that does seem to be the terminology people use. Anyway, writing this document, and thinking back on the long and winding history of this project, I feel a combined sense of sadness and pride. If this sounds pessimistic, don’t worry, it ends optimistically.

The sadness I feel for what might have been. I designed a lot of different games both in my head and in prototype form in the last year. Some of them were interesting, Killer Riff especially, and it’s a very deep rabbit hole to imagine what would be different if I had stuck to one of those concepts.

Good news, of course: I can revisit these ideas later! But there is only one thesis, and the other reason I feel sad is that it would have been nice to deliver a fully mature and highly-polished project, in a way that would only have been possible if I had decided much earlier in the process exactly the nature and scope of what I would make, so I would have the time to make it. Working with someone else might have facilitated that, but even if I had just been able to spend the entirety of the second semester polishing something from the first, that would have made a profound difference.

Even in the case of Rifftide, I had many exciting ideas for deep applications of that gameplay, but a shortage of ideas or enthusiasm has never been the problem. In the end, I must of necessity deliver something that is much smaller-scale and less “done”

than I wanted, and then agonize over its shortcomings. I realize this is a perfect description of any creative work that actually makes it out the door, but I still feel sad.

I also feel proud. I wanted to create a new kind of music game, one that turned sheet music into a world to explore in an exciting way, and I did. I wanted to compose music and use that as the basis for gameplay, and I did. I wanted to give players the feeling of making their own music, and (according to the playtesters at least), I did. I wanted to pursue the idea I loved, and I did.

I know of no other game like Rifftide, no other experience that interprets music in this way. Whatever criticisms may be leveled at the final product, however much more I could do to expand and improve the game, it stands as the successful fulfillment of my thesis. What greater validation is there?

See, I told you it ended optimistically.

VI. Works Consulted

The 5,000 Fingers of Dr. T. Dir. Roy Rowland. 1953. DVD. A fantastic re-imagining of music lessons as a literal prison run by a diabolical instructor.

Abramovitch, Seth. "David Mamet's Master Class Memo to the Writers of The Unit." *Movieline*. N.p., n.d. Web. Nov. 2013. <http://movieline.com/2010/03/23/david-mamets-memo-to-the-writers-of-the-unit/>. Some useful words from David Mamet on narrative structure.

Bad Hotel. Vers. IOS. N.p.: Lucky Frame, 2012. Computer software. The player builds a hotel room by room, and the different rooms and overall structure generate the soundtrack dynamically.

Blow, Jonathan. *Braid.* Vers. OS X. N.p.: Number None, Inc. / Microsoft, 2008. Computer software. A superb example of level design, using the same game mechanic for wildly different purposes in each level.

Bogost, Ian. "1. Procedural Rhetoric." *Persuasive Games: The Expressive Power of Videogames*. Cambridge, MA: MIT, 2007. N. pag. Print. The first chapter makes the case that games are representational systems that can be used to make an argument.

Cavanagh, Terry. *Super Hexagon*. Vers. IOS. N.p.: Terry Cavanagh, 2012. Computer software. An example of synaesthesia/sync gameplay.

Cavanagh, Terry. *VVVVVV*. Computer software. Vers. OS X. Terry Cavanagh, 2010. Web. A superb example of level design, using the same game mechanic for wildly different purposes in each level.

Child of Eden. San Francisco, CA: Ubisoft, 2011. Computer software. An example of synaesthesia/sync gameplay.

Coulson, Denver. "Designing Without Words." *Gamasutra.com*. N.p., n.d. Web. Nov. 2013. <http://www.gamasutra.com/blogs/DenverCoulson/20131115/203952/Designing_Without_Words.php>. A major inspiration for the tutorial design, promoting minimal reading.

Crypt of the NecroDancer. Vers. Steam. N.p.: Brace Yourself Games, 2013. Computer software. An example of mimicry: rhythm/dexterity gameplay.

Dance Dance Revolution. Redwood City, CA: Konami of America, 2001. Computer software. An example of mimicry: rhythm/dexterity gameplay.

Elite Beat Agents. Vers. Nintendo DS. Nieuwegein: Nintendo, 2007. Computer software. An example of mimicry: rhythm/dexterity gameplay.

Epic Mickey. Vers. Nintendo Wii. Burbank, CA: Disney Interactive Studios, 2012. Computer software. An example of environmental manipulation gameplay.

Gage, Zach. *SynthPond*. Vers. IOS. N.p.: Zach Gage, 2008. Computer software. An example of music manipulation/creation.

Gee, James Paul. "Chapters 1-3." *What Video Games Have to Teach Us about Learning and Literacy*. New York: Palgrave Macmillan, 2003. N. pag. Print. The first few chapters examine how games teach literacy of their systems through the user's interaction.

Harmonix. *Amplitude*. Vers. Sony PlayStation 2. Foster City, CA: Sony Computer Entertainment, 2003. Computer software. An example of mimicry: rhythm/dexterity gameplay.

Harmonix. *Frequency*. Vers. Sony PlayStation 2. Foster City, CA.: Sony Computer Entertainment, 2001. Computer software. An example of mimicry: rhythm/ dexterity gameplay.

Harmonix. *Guitar Hero.* Los Angeles, Calif.: Activision, 2008. Computer software. An example of mimicry: rhythm/dexterity gameplay.

Harmonix. *Rock Band*. Redwood City, CA: Electronic Arts, 2007. Computer software. An example of mimicry: rhythm/dexterity gameplay.

Iwai, Toshio. *Electroplankton*. Vers. Nintendo DS. Redmond, WA: Nintendo of America, 2006. Computer software. An example of music manipulation/creation.

Karaoke Revolution. Redwood City, CA: Konami of America, 2003. Computer software. An example of mimicry: pitch gameplay.

The Legend of Zelda: Ocarina of Time. Vers. Nintendo 64. Redmond, WA: Nintendo of America, 1998. Computer software. An example of music manipulation/ creation gameplay.

Loew, Karen. "Squiggles Are the New Quarter Notes: Why Music Looks Different Now." *The Atlantic*. N.p., n.d. Web. Nov. 2013. <http://www.theatlantic.com/entertainment/archive/2013/10/squiggles-are-the-new-quarter-notes-why-music-looks-different-now/280271/>. An example of an alternative form of musical expression.

Luigi's Mansion. Vers. GameCube. Redmond, WA: Nintendo of America Inc., 2001. Computer software. An example of environmental manipulation gameplay.

Okami. Vers. Sony PlayStation 2. N.p.: Capcom, 2006. Computer software. An example of environmental manipulation gameplay.

"Piano Projections Help You Play a Tune." *YouTube*. YouTube, 02 Oct. 2013. Web. Nov. 2013. <http://www.youtube.com/watch?v=DOSEekGuq0w>. An example of an alternative form of musical expression.

Rez HD. Vers. Xbox Live Arcade. San Francisco, CA: Sega, 2008. Computer software. An example of synaesthesia/sync gameplay.

Rocksmith. Montreuil-sous-Bois: Ubisoft Entertainment, 2012. Computer software. An example of using a game to teach music knowledge.

Singstar. Foster City, CA: Sony Computer Entertainment America, 2007. Computer software. An example of mimicry: pitch gameplay.

Stinson, Liz. "Conductive Ink Turns Paper Into Musical Instruments." *Wired.com*. Conde Nast Digital, 16 Oct. 0013. Web. Nov. 2013. <http://www.wired.com/design/2013/10/conductive-ink-turns-paper-into-musical-instruments/>. An example of an alternative form of musical expression.

Studiobean. *Soundodger*. Computer software. *Adultswim.com*. Studiobean, 11 Oct. 2013. Web. <http://games.adultswim.com/soundodger-puzzle-online-game.html>. An example of synaesthesia/sync gameplay.

THREAKS. *Beatbuddy*. Vers. Steam. N.p.: THREAKS, 2013. Computer software. An example of music manipulation/creation gameplay.

Trebella, Whitaker. *Pivvot*. Vers. IOS. N.p.: Whitaker Trebella, 2013. Computer software. An example of synaesthesia/sync gameplay.