Victor Ivanov
40031772

Qurotema - Report

Over the last 4 months, I've dedicated myself much more seriously to a project that I've wanted to finish for over a year now. *Qurotema* is a first person game set in a dark surreal and surreal environment, focusing on music production and exploration of mechanics.

Media

Video of a full playthrough.
https://www.youtube.com/watch?v=pqO7EeNAWfk&feature=youtu.be

Build (Windows).
https://v-os.ca/media/downloads/qurotema/qurotema_windows.zip

Goals

I had a few goals in mind when starting development on *Qurotema*.

1. To create a game where players would have little to no instructions as to how to interact with the world, and would need to experiment on their own to discover their ability to influence the game world.

2. To create a game that gives players the ability to influence, both directly and indirectly, a dynamic soundscape.

3. To create a game that feels aesthetically refined, making for beautiful picturesque screenshots, as well as making basic systems like camera movement and player navigation into beautiful experiences.

4. To tell a portion of a story that revolves around communication, misunderstanding, balance, and curiosity.

For the most part, I feel like I've accomplished these objectives. *Qurotema* is a game whose visuals and audio, I hope, speak for themselves through the video provided above. The most questionable objective is the fourth, seeing as how the narrative is currently too shallow for my liking, especially the ending of the game. I feel that adding more narrative content and integrating it more deeply into the game (voice acting? in-world interface for the text? better audiovisual feedback for text?) would make a great difference.

What I am far more interested in, however, is what I've learned over the course of this project. Achieving these goals is, of course, important, though it's much more exciting to look back on the development and reflect on the unexpected parts of a project. Otherwise documentation would consist of nothing more than "I had an idea, then I made it".

What I'd like to speak about in this report is a collection of the more remarkable lessons I've come across these past 4 months. These lessons are:

1. The effect of prototypes on creativity.
2. Aesthetics-centered games.
3. Technical challenges of project development.
4. Making connections between multiple projects.

The effect of prototypes on creativity.

Throughout my education, I've been told that prototyping is incredibly useful because it allows you to iterate through different ideas, to develop solutions to problems, and to essentially try different things quickly. I agree with all these arguments, but working on *Qurotema* gave me some insight on another benefit I believe I overlooked. This benefit has more to do with how having some sort of prototype made a positive change in regards to my creativity.

In the beginning, I had an idea for one of the instruments; the strings. I also knew that I wanted to have multiple instruments, had no concrete ideas for what those other instruments would be. I decided it would be best to focus on other things until those ideas come up, and this was a good move.

Further into development, I was busy making the sound system. As I was finishing that system, the idea of the rings came to me quite naturally, and the sequencer instrument followed shortly.

I later realized that the sound system itself played a role in my ability to come up with these ideas. The fact that it existed effectively defined a set of limitations for what is possible, and set up a foundation for me to build ideas on. Limitations can be a useful tool for creativity as they give you less opportunity to explore too chaotically in a wide territory of ideas.

Although the sound system was not a prototype persay, it was a base for these ideas to come to life. So, I know that in future projects, if ever I'm having trouble coming up with creative ideas, creating the system(s) that they will be based on can have a very positive effect on how easy it is to come up with those ideas in the first place. Though, I do need to watch out and not over-commit to this idea, as it's possible that I get better ideas down the line, and need to remake those systems. All things in moderation.

Aesthetics-centered games.

Qurotema is not a standard game, in that it doesn't feature typical objective-oriented gameplay. A big focus in *Qurotema* is aesthetics. What I had to consider at the start of this project was "what mechanics are better suited for an aesthetically-oriented experience?".

In any given game, we want tight controls, responsive cameras, clear instructions, minimal visual clutter, and so on. Essentially, the goal is to allow the player to engage with the game

more seamlessly in order to partake in the fun challenges of the game. The reason why tight controls are a great thing is because they allow us to face greater challenges with less frustration. If we are not confidently in full control of the character, then every loss feels like an unfair punishment. Why should we be blamed when the control is not fully ours?

This changes when the types of challenges change. In *Qurotema*, there is no platforming, or shooting, or timed events. The challenge is to explore and uncover systems in the game, and to come across beautiful experiences. That doesn't mean that we can do whatever we want and call it "good gameplay", but it does mean that we need to redefine the nature of our mechanics to suit the challenges.

For instance, if we want the player to uncover different modes of interaction, mapping the toggles for those modes on the mouse makes a lot of sense, since players will likely click the mouse buttons just to see what they do. Assigning mode changes to, say, the letter "J" on the keyboard would be a bad idea, since the player probably doesn't have the intention of pressing every key to see what it does.

So, controls in *Qurotema* are mapped to unique keys: mouse buttons, shift, alt, WASD, and space. "Alt" is the least likely key to be pressed, and it enables flight, which is a pretty cool reward for finding it. So there's an attempt at recognizing that the better rewards should be hidden behind bigger challenges.

Another example is the camera. *Qurotema* has a signature camera shake that is at its most extreme when the player is standing still. This camera shake can be extremely annoying in, say, a first person shooter. Not to mention, everything from camera to player movement is smoothed. I think it's fair to say that even in *Qurotema*, when interacting with some instruments, the smoothing and camera shake can get in the way.

However, the other 'challenge' of the game is to find beautiful experiences. Maybe finding a monument is a cool moment because it gives you something to focus on in the landscape, making a beautiful composition? But this beauty is emphasized by the slowly panning camera, and the smooth player motion. Imagine instead tackling the 'challenge' of finding beautiful moments with an extremely snappy camera. It's practically impossible to create the smooth panning motion with that system, and one might become frustrated because the mechanics don't interact with the challenges in a way that's enjoyable for the player.

I think realizing this has helped me immensely with understanding design on a more intimate level. I won't sit here and lie: in the beginning, a lot of what I was making, I was making because I just thought it looked cool. This is a valid reason, of course, but having a grasp on *why* it looks cool, and *why do we care* that it looks cool can bring a product to the next level with more deliberate decisions.

Technical challenges of game development.

Throughout this project I came across many issues. How do I make a custom skybox? Does the engine support X feature? Why is my game performance bad in this particular situation? Do I need to change engine version to get Y feature?

I've come to realize is that it's a miracle anything works. The concrete experiences I had with this project are unique to Unity, although they are manifestations of trends I've recognized in other engines and tools, too.

In short, these are behemoth pieces of software with huge teams maintaining them, and the bugs and issues are bound to be endless. This idea solidified when I learned that enabling v-sync would sometimes cause what I consider to be fairly noticeable stuttering in a build. 7 pages were filled on a Unity forum thread over the course of 2+ years about people being upset about this, and wondering how this could have gone unnoticed. I had this issue, and I am yet to find a proper fix for it. Yet, thousands of Unity games are made every year, some of which have these issues in them.

It's given me a better perspective on what to expect from using these tools, even when moving on to a more professional level. These problems will exist with anything of this scale, and I need to be prepared to face them.

Making connections between multiple projects.

As is commonplace in my work, I try to connect projects together though an overarching narrative by sprinkling elements of a shared universe between all of them. In the case of *Qurotema*, I decided the best way to do that would be with i-tema, the fictional language I've constructed.

The language is currently in-development, and documentation will be posted on my site once that's done. For the extreme minority of players who are interested in these types of things, I hope they will feel motivated to explore further and migrate from *Qurotema* to my site. If that happens, I'd like to keep exploring ways in which different products can act as interconnected pieces of a larger narrative.

This idea fascinates me as it acts almost as a justification for each product: it is a representation of a point in time of a larger story. In addition, it helps future stories as it provides a world-building-friendly narrative universe to draw inspiration from. I am excited to fully integrate *Qurotema* into the story of my website and grow the narrative.

Shortcomings

I'm very happy with *Qurotema*, but I'd be lying if I said it's perfect. There remains some work to be done to polish the project up. Among other things, animations with Monoliths are lacking in the smoothness everything else has, and the story is what one might call 'incomplete'. These are not deal-breakers, and I intend to fix them in the coming weeks. Nonetheless, *Qurotema* is in a much better state than I imagined it would be at this point in time.

Bringing things back to the first lesson on prototypes, I feel I could have done even more if I knew the speed that a finished foundation would bring to my project. I did a very large amount of work in the past few weeks on *Qurotema*, and for that reason I feel my overall productivity could have benefited earlier on if I focused on completing fundamental systems immediately.

Qurotema has taught me much, and I hope to bring these lessons with me on my next project!