**TEMASEK POLYTECHNIC**

**SCHOOL OF INFORMATICS & IT**

**DIPLOMA IN GAME DESIGN & DEVELOPMENT**

**AY2023/2024 OCTOBER SEMESTER (LEVEL 2) TERM A**

**GAME MATH AND PHYSICS (CGE2C15) TERM A**

**Project Documentation (10%)**

|  |  |
| --- | --- |
| **Class** | P03 |
| **Name as in register** | Low Tong Ee Thom |
| **Group #** | **5** |
| **Topic covered** | Particle system |
| **YouTube URL** | Make sure the link is shareable! |

**You must submit:**

* **A discussion about the** **Technical Problems & Solutions encountered**
* **A reflection about this assignment, and also of GMAPS.**

**Each question is on a separate page below.**

1. **Technical Problems & Solutions**

**Discuss the *technical* problems you faced while writing your code, and how you overcame them.**

**Technical problems do NOT include problems you might have had with time-management, knowledge of basic GMAPS topics, illnesses, part-time work, broken computers, lost files, or other distractions, etc.**

**Focus on**

* **Problems you had in understanding concepts related to the chosen topic, and how you managed to gain a better understanding, e.g., via online references, use of ChatGPT and other AI, etc. Make sure you list all references used.**
* **Problems you had in coding your demo implementation, and how you managed to overcome these. Give specific examples.**

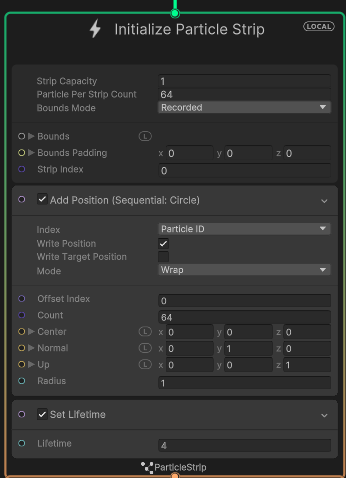
**Introduction:**

As this is my first time working with VFX graph, I find hard to understand on how to even make particles. So, when I first started out, I decided to have a road map to know where to start.

1. Try to create a single strand of Grass.
2. Making a patch of grass
3. Adding additional features.

**Making Grass:**

The most difficult thing was getting the grass to be rendered as a particle. There were only two ways to render the particle. Individual particle or as a particle strip. I understood how to render using individual particle, but I find it hard to render them together using the strip.



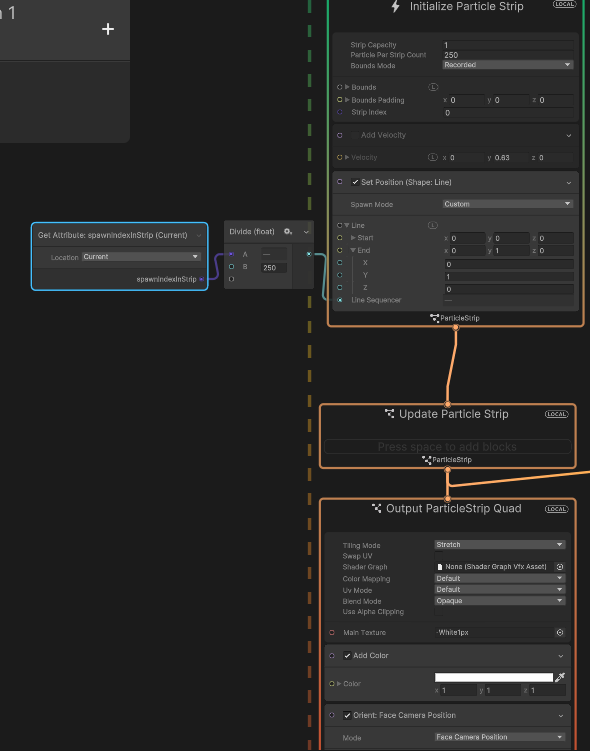
At first, I could not figure if the particle strip position is added to either the particle strip or the individual particles in the strip.

Even after reading the [documentation](https://docs.unity3d.com/Packages/com.unity.visualeffectgraph@12.1/manual/Block-IncrementStripIndexOnStart.html?q=particle%20strip), it still does not give a good idea on how the particle strip works. In fact, there is nothing stated on how to use particle strip on the unity documentation.

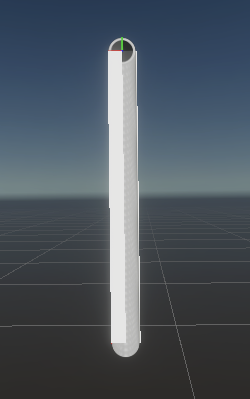
***How I solve it:***

I did quite a bunch of research on how to render particle strip. What help me was the 6-part tutorial video series by [Thomas Iché](https://www.youtube.com/@thomasiche8734/playlists). Where it the 4th and 6th tutorial teach how to use particle strip for visuals like lighting and hair.

When spawning the grass, the particle strip will apply changes to every particle assign to that strip. So, the changes will be applied to every particle in the initialize particle strip. The mesh will then be made as Unity connect the particles together to form a **single quad** which can be used to apply the texture.



How a sample visual script look like to render a line using particle strip



How the strip would look like in Unity

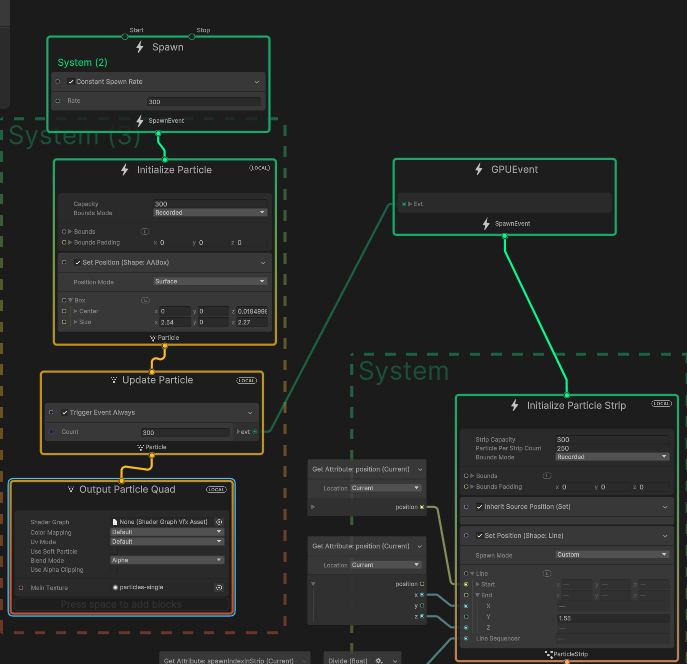
**Figuring out how to render a patch of Grass**

The next difficulty come which is how to render particles in a particle? Throughout the documentation, there were nothing about being able produce more particles from existing particles. So, I need to know,

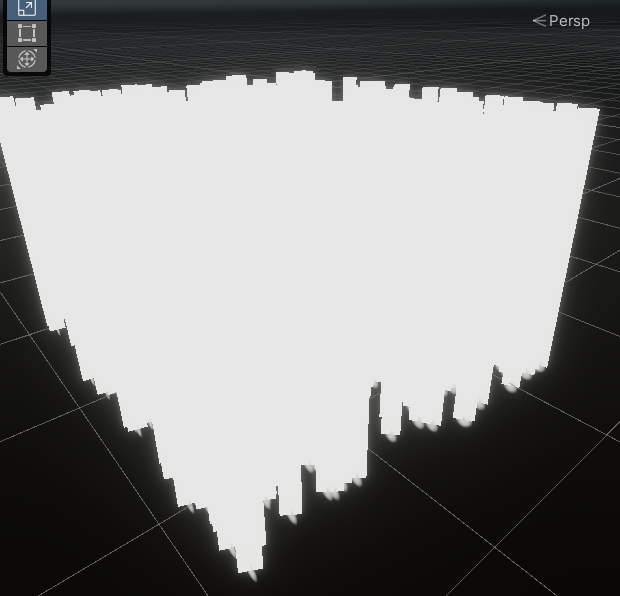
1. How to have particles to throw out more particles (or particle strip)
2. How to group up this particles and particle strip to form a terrain

**How I solve it:**

Thanks to [Thomas Iché](https://www.youtube.com/@thomasiche8734/playlists) , his 6th tutorial introduce GPU events and how to utalise them. It was not stated in the unity documentation since it is considered an **experimental block. GPU events** allow particles to throw out event in the GPU to signal other listeners like an observer pattern. This makes particle call out events to start spawning other particles.



A simple script to make a square patch using particle system



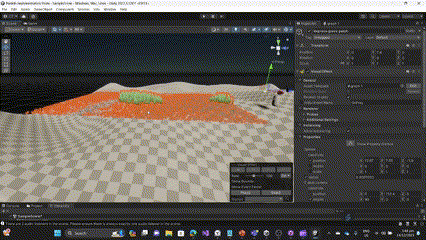
The end result is a square patch using GPU event.

**Rendering the grass in a terrain:**

Next is how the grass should render in a terrain. There weren’t a lot of examples I could reference to learn how to make particles spawn in a randomly procedure terrain. Another problem was that VFX graph do not support spawning attributes that are varying shapes like a terrain. The only spawning attributes VFX graph that does that is through reading a noise map which I can’t use since the terrain and VFX graph use different noise map. This was by far the hardest problem I had when trying to render the grass.

**How I solve it:**

While brainstorming, I realise that, if the particles were to collide with the terrain, it would know where the collision happen in that point of time and would spawn the grass strand. After much research, I get to know about “[collide with depth buffer](https://docs.unity3d.com/Packages/com.unity.visualeffectgraph@10.2/manual/Block-CollideWithDepthBuffer.html)” attribute that VFX graph provides. Where it will stop the particles when it see a visible object in the scene. With this knowledge, I can set the collision to kill the particle the moment it hits the terrain surface. I can the send a event to spawn the particles the moment it dies. That way, the grass will know where to spawn in the terrain and it will randomly generate it by itself.



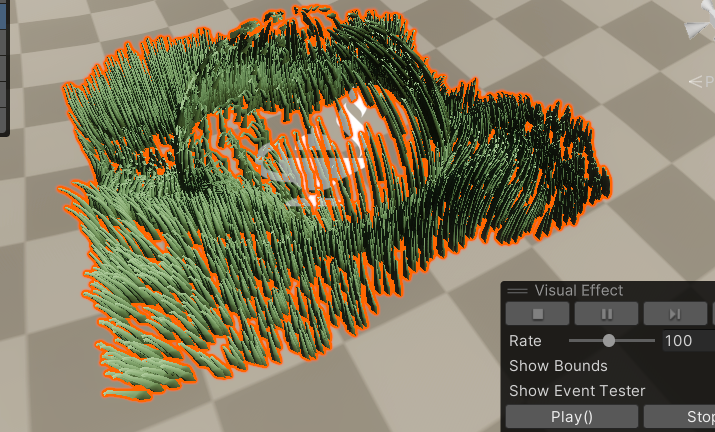
How the grass production works

**Adding (additional stuff) physic interactions:**

I wanted to be more ambitious for this project and I thought it would be cool to have some physic interaction to the grass. The problem was figuring out how to add this physic interaction to the grass patch.

**Adding collision to the grass patch:**

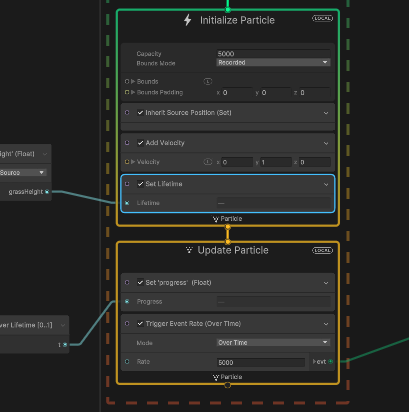
While trying to add collision to the grass, I had a lot of issues with setting it up due to how my grass was set up initially. My grass initially would wrap around the grass because the particles in the grass strand will move further away from the each other as it is rip apart by the sphere collider.



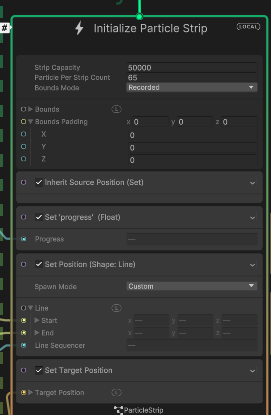
An example of the problem I had initially with the grass

**How I solve it:**

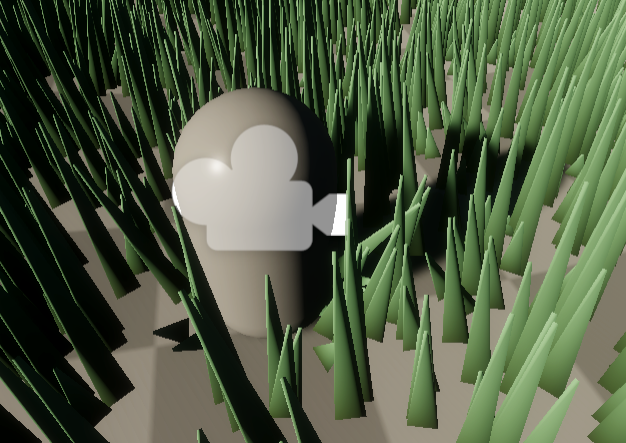
Initially the grass patch strip mesh would be produced due to the velocity of the particles that is shot up. Which cause it to then wrap around the collider as it tries to avoid hitting it. After much trial and error, I manage to make the grass strip mesh to be produce based of that line, this prevents the grass from wrapping around the collider as a result.



Before improvement (where the grass patch would wrap around the collider)



After improvement (where it would maintain structure no matter what)



Now the grass would not wrap around the sphere collider

1. **Reflections**

**Reflect on:**

* **Your project, including the presentation and implementation parts. Discuss how useful you found the project work (e.g., relevance to game development, development of your technical knowledge, development of your programming skills, etc.) and how you approached it (e.g., your attitude, self-discipline, independence in learning, etc.).**
* **GMAPS as a whole. You can also use the discussion points suggested above here (e.g., relevance to game development, your attitude, etc.).**

1. **Project Reflection**

**Before embarking on the project:**

Initially, I didn't consider the particle system to be particularly special. In my mind, particles were just basic elements like sparks or snow. The sole reason I selected it was due to its perceived versatility compared to other topics.

**During the project:**

Upon delving into the potential applications of the particle system during my research, I was astounded by how much I had underestimated its capabilities. It proved instrumental in animating objects, simulating living creatures, and infusing vitality into foliage.

One memorable instance involved a slide featuring a monkey, a creation I spearheaded because, at the time, I believed every grass strand and object had to be a distinct game object. It was a revelation to realize that certain game objects could masquerade as particles. The concept of particles colliding with game objects to emulate the behavior of a game object without revealing its true nature also fascinated me. This project provided valuable insights into how games achieve intricate details in foliage, enemies, and miscellaneous objects.

Throughout the project, I gained a solid understanding of and proficiency in using Unity's VFX graph. However, the learning curve was steep, particularly since it was my inaugural experience with visual scripting. Despite the challenges, the process became enjoyable once I achieved the desired results. Multiple trials and errors were necessary for both satisfactory outcomes and optimization to render more realistic grass strands.

**Post-project reflections:**

Completing this group project left me feeling gratified and enlightened. It broadened my understanding of how particles can be applied in diverse ways beyond their apparent simplicity. Additionally, I emerged with newfound expertise in utilizing the VFX graph, a valuable asset for my future game development projects.

This project also served as a catalyst for exploring new possibilities. Recognizing the limitations of VFX graph due to Unity APIs, I pondered the prospect of crafting my own particle system implementation within Unity. Considering the feasibility of coding it using compute shaders and noise maps, I see it as a promising and enjoyable side project for my upcoming holidays as I delve deeper into particle system intricacies.

In conclusion, the project was both enjoyable and fulfilling, equipping me with valuable knowledge about utilizing and implementing particle systems in game development.

1. **GMAPS reflection**

**What are my thoughts on this subject?**

I have a great affinity for this subject. I eagerly anticipate each worksheet, finding joy in completing them. My only gripe is the lack of more challenging questions on each worksheet. The knowledge acquired from GMAPs has provided me with a practical understanding of applying these concepts in gaming, such as using matrices for rotation. It also excels in simplifying intricate concepts, like physics formulas, broadening my comprehension of their practical applications. The worksheets and projects have fostered my self-directed learning, a unique feature compared to other year 2.1 subjects. GMAPs strikes a balance, offering structured learning while allowing exploration of advanced concepts beneficial to my game development journey.

**Did the subject benefit me?**

The subject has directly and indirectly contributed to my growth. It enhanced my understanding of various physics and math topics and sparked my curiosity for delving into more advanced concepts. As I plan to undertake several side projects during my holiday, GMAPs serves as a wellspring of ideas for creating game engines or math libraries.

It also indirectly influenced me in adapting to new technologies. Engaging in the group project forced me to swiftly learn the VFX graph within weeks, prompting me to explore a new scripting and behavioral approach. Juggling multiple commitments, I learned valuable lessons about effective learning strategies. For instance, rushing through tutorials and skimming documentation, though time-efficient, led to overlooking crucial information. Reflecting on my mistakes, I recognized the importance of thorough understanding for effective development.

**Conclusion**

In summary, this subject is immensely enjoyable and intriguing. Even before the semester commenced, my anticipation for delving into math and physics for games was palpable. The insights and knowledge gained have significantly shaped me as a programmer. I eagerly await future opportunities to apply these formulas in upcoming games!