MAJOR PROJECT

PART 2

Report on SMii7Y Gang vs. Caldera

Adeel Zafar (1805187)
Module Year: 2022/3

TABLE OF CONTENTS:

1 Introduction
1 Modelling the Hero Nerf Asset
2 The Environment
4 Motion capture/animation
6 Exporting
7 Animations and Sequencer
8 Filmography
9 Other Technical Hinderances
10 Conclusion
11 Credits

INTRODUCTION

The project dubbed Back-2-Verdansk has relatively seen as expected highs and lows mixed with tedious work. From mostly proficiently well off work that has gone to periods of tedious work given the nature of replicating the same workflow for all 4 characters to periods that has almost jeopardized the entire project, this is the in-depth look into the workflows used and problem solving situations dealt with when making the animated short 'The SMii7Y Gang vs. Caldera'.

MODELLING THE HERO NERF ASSET

Given the time constraints and focus on animation and scene setup, I resorted to focusing on only one manually modelled object, which is the Nerf Stampede that the Carver Butcher enemy uses to "kill" Puffer as an irony point opposed to the other members in the gang being killed by actual guns given the nature of Caldera's wild nature of guns.

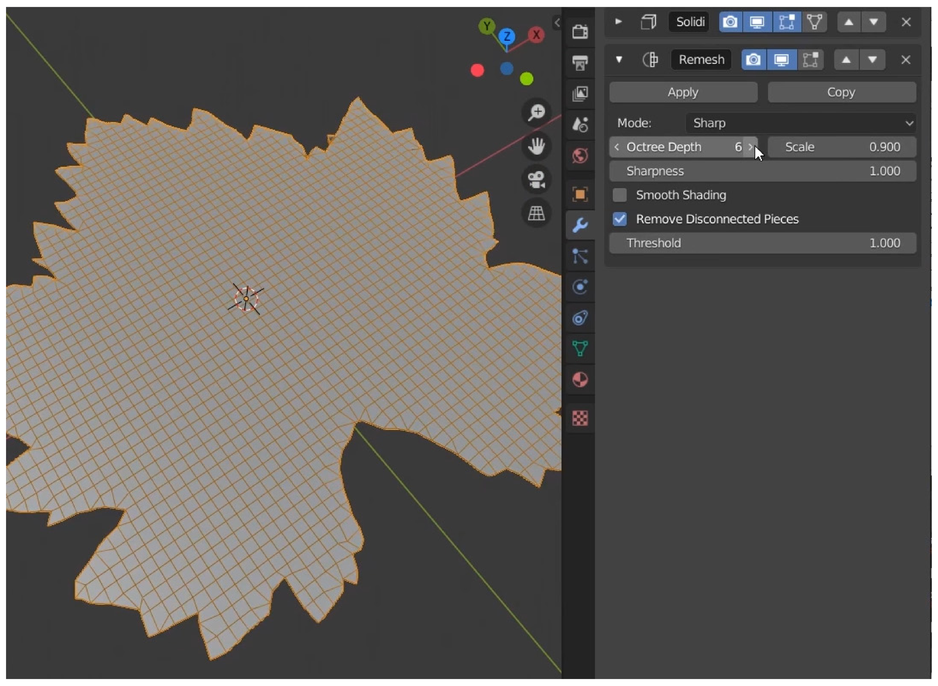
The first step was photographing every angle of the gun for detailed referencing and even some textures to be used such as symbols, text and the likes. I then laid out everything in PureRef that makes for a great collage and reference photo so I can zoom in and out throughout the many reference photos used.

Then was a matter of blocking out the major components from the body, barrel and trigger guard to detailed pieces for each component. Hard surface modelling with a focus on game ready polygon workflows were incorporated given the nature to utilize baked normal maps later on. Parts of the gun were modelled based on a combination of side referencing, close up shots and eyeballing, as such its not 100% accurate to the actual gun's dimensions, but mostly makes sense given its not meant to be mathematically correct as to a more CAD like workflow.

After a week of tedious individual parts modelling, a quick UV unwrap was done for Substance Painter texturing. Thanks to the simplistic nature of the gun and the mostly new look, the bulk of the work was on making a color ID mask in Blender by assigning faces to their respective color and baking that for Substance Painter. For the most part one material with a corresponding color was used for each. For bevel detail, a bevel shader was used in Cycles and baked for Substance Painter too. Once done exported as a high resolution 8K map for the main body and a 4K for the barrel.

THE ENVIRONMENT

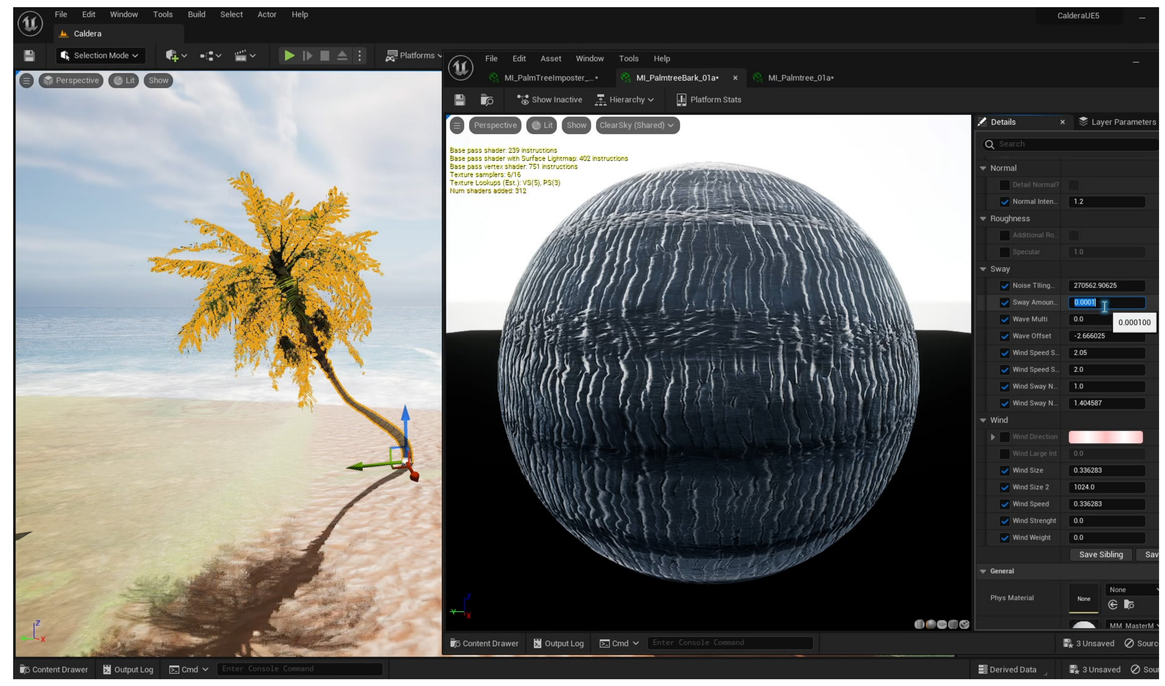
The first step was the landscape that was simply carved out from the surrounding location of Runway to Peak, just selecting the land, I then sent the landscape only as a png to Blender with the Leafig addon that creates a cutout from the image without having to go through the hassle of turning it into a path and convert to mesh. Obviously given the addon having to make a cutout with non-uniform topology and triangles, I took the mesh into a remesh modifier with a fairly high poly count to make it square polymesh based so I can then start sculpting. For the most part I sculpted based on the topology hinted on the map and from game reference footage and used the Corsair fighter plane as reference since it was the easiest to scale to the map given the runway and hanger sizes already given.



Using Blender Secret's trick of using a solidify modifier before the remesh modifier for a flat plane/cutout.
https://www.youtube.com/watch?v=HvHqa-ajWco

Once a basic enough sculpt of the landscape was done, I converted it into a height map by simply rendering the z-depth pass from a top down camera as an EXR, then it was just a matter of importing into Unreal Engine. Given how much more proficient the landscape sculpting tools in Unreal Engine were, I took it from there to fine tune the landscape before starting to brush out foliage.

For the foliage, I choose to use a great Tropical tree, bush and grass collection by Dekogon Studios that also had the trees swaying that really helped sell the coastline windy shore setting. Although quickly I found that the trees used a position offset system for the tree swaying that caused it to multiply in speed when 10000 meters away from origin. To solve this, I simply turned down the sway amount values in the material as suggested in the FAQ section on the product page.



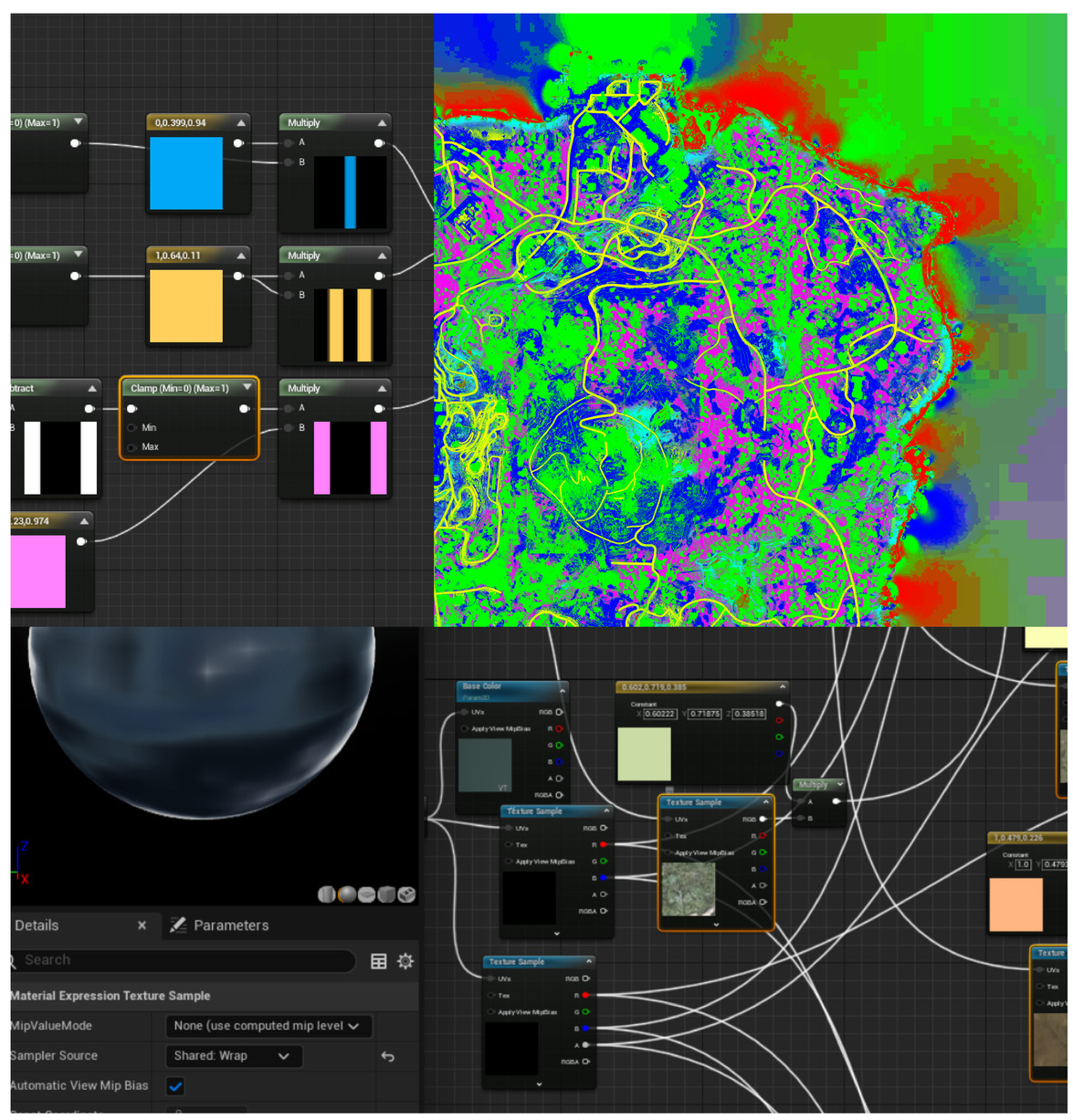
Adjusting the sway amount in both the trunk and leaves materials.

https://www.unrealengine.com/marketplace/en-US/product/tropical-foliage-and-landscape/questions?sessionInvalidated=true

With that solved, I then started painting away, although quickly found even with the density value turned down for the trees that it was too much when combined with the smaller plants. As such for control, I decided to paint in stages starting with just the smaller grass blades and rock that shouldn't interfere with the characters and mostly there to cover up the flat grass textures as micro detail. Then I went with the bushes and smaller trees and finally the large palm trees. Another benefit for choosing Unreal Engine was how nanite helped greatly to reduce the LOD pop-in issue with the further away palm trees that distinctly didn't look good with a blobby/low res look just 100's of meters away. So having to click two checkboxes opposed to finicking with the LOD to make it less obviously low res was greatly appreciated.

The landscape material was simply made by first making a mask to distinguish the 6 different materials, including base tropical green grass, a more dead type of grass, cliffside rocks, road dirt, beachside red sand and rocky red sand. For this I initially went with covering RGB then hoping I can make use of a node system like this for making use of the CMY channels.

Although unfortunately with that node setup and my lack of the Unreal Engine material node system, I fell through that plan and decided on making two color ID masks, one that used RGBA to cover the first 4 channels and a second with just red and blue for the last 2. The end result was a very spaghetti looking material setup since I had to mix between 6x materials consisting of base color, roughness and normal textures. Only issue was going past 16 textures tripped the texture samplers limitation, which thankfully was solved by just switching the sampler source for each texture to 'Shared: Wrap' as a bypass.



https://forums.unrealengine.com/t/how-can-i-extract-any-colour-from-a-texture-map-inside-of-a-blueprint/287547/9

https://forums.unrealengine.com/t/texture-samplers-limitation-is-still-present-in-4-9/42958

Finally the sky with dynamic clouds was simply provided with the Tropical Foliage and Landscape collection alongside other collections like Modular Cliffs by CGMontreal, Brushify - Cliffs Pack and Quixel Megascans for rocks that were both in small scale for foliage brushing and larger modular pieces placed alongside rocky cliffsides and beachsides.

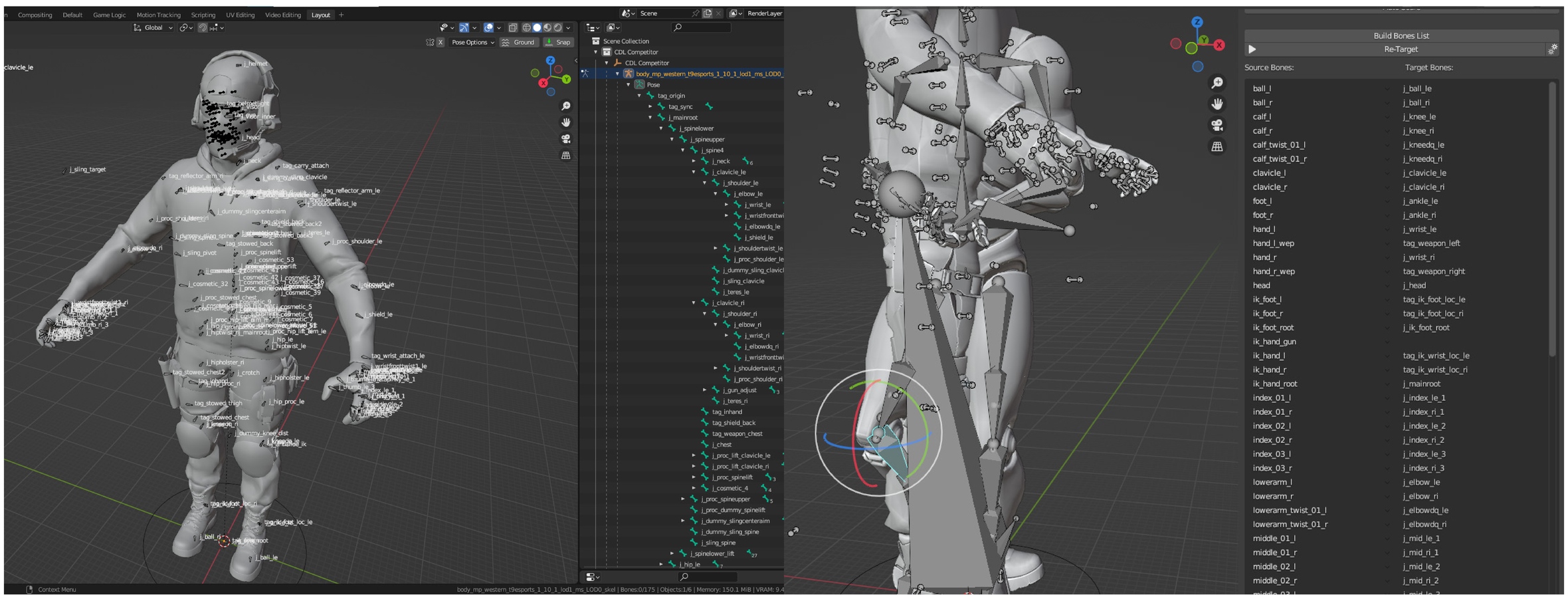
MOTION CAPTURE/ANIMATION

The big unfortunate to the project is the technical issues we had with Shogun Live, mostly down the no subject issue. This was a case where the program's preferences disallowed auto-detecting a subject when presented with 63 out of 63 motion tracking points and double checking it was in the right position/setup as per the Vicon guide. Given other complications with the system, I had to decide on ditching motion capture data and rely on third party sourced motion capture animations.

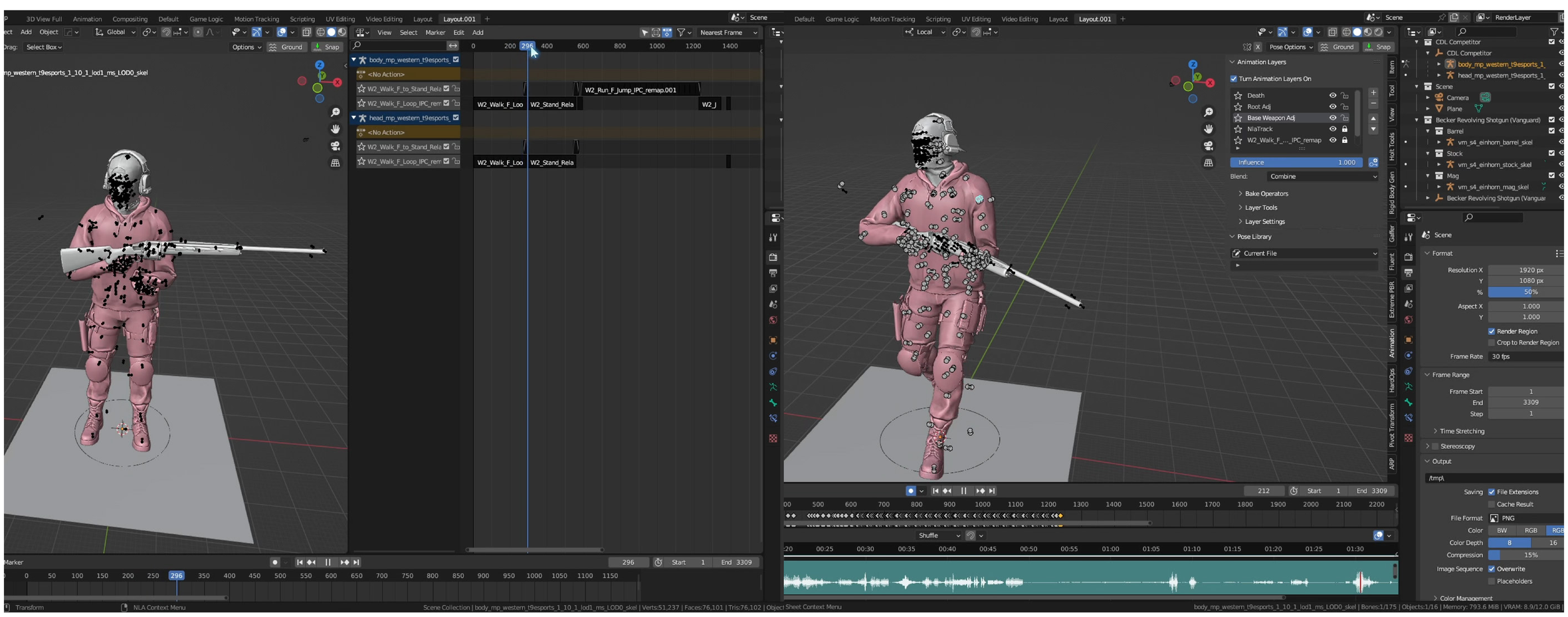
For this I decided on two expansive animation packs that fits with the soldier rifle poses that are game ready given the choice for in-place animations which will be seen effective throughout the project. Although given the two different publishers, retargeting will need to be rechecked for the other publisher's animation packs that's comprised of falling/bumping animations.

The focus of the animations coming from the rifle mocap pack was a great choice thanks to how the animations where labelled based on 4 speeds, stand relaxed, walk, jog and run, and direction such as _F being forward, _B being backwards and turning directions like _L45 meaning left 45 degrees and even all the way to 180 degrees. With this expansive per direction and speed variety, finding the fitting animations for each character based on whether they stand relaxed then immediately start running like SMii7Y, stand relaxed to jog and stand aim like with Blarg when he aims and shoots the Bazooka, or even immediately do a 180 like with Puffer when he hears the first bullet whizz. Additionally it comes with in place versions perfectly looped so minimal blending when repeating animations was needed.

In terms of applying the animations onto the Call of Duty Modern Warfare (2019) armature system, it was as simple, although tedious, of going through and checking each bone correlated to each other like how Rifle MoCap pack called the shoulder instead of upperarm and thigh instead of hip, etc. Auto-Rig Pro's remap (retargeting) autodetect feature didn't detect many matching bones, rightfully so given the vastly different naming schemes, so it was a matter of going back and forth double checking the bones correlated. Although once done, you had the grateful choice of exporting the bone list as a preset, and given all the other 7 characters share the same armature system since they share the same game engine. On first try, everything was rotating right with no obvious issues, only thing was the fact that their arms went way to inwards into the body and the inconsistent hovering above ground issue which can be solved with animation layers.

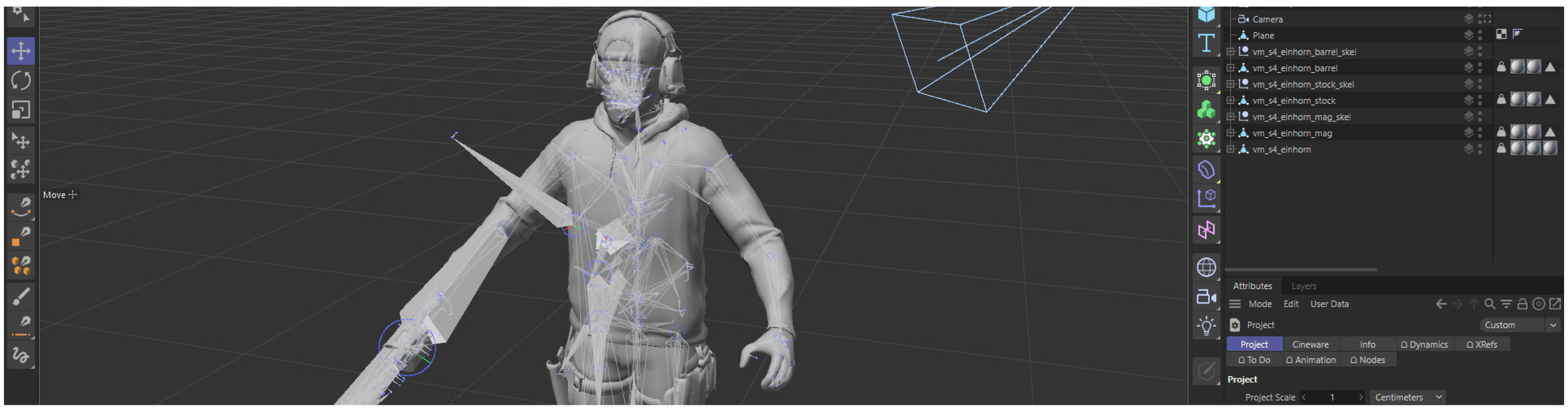


Although before that was done, I decided to do the base blending and repeating of clips in the NLA editor so that animation layers weren't interfering, all of which was timed with key points determined by the sound composition like when the munitions box break scaring the 4, the first bullet whizz and indicative stances based on their dialogue. Once blended, animation layers came in to save the animation by first importantly adjusting the arms to correctly hold their respective gun. Although this proved tedious as their left arm holding the bottom barrel of the gun was constantly drifting away, so I'd have to adjust the arm constantly and delve into places where I can copy paste keyframes or succumb to manual work. The second major animation layer was solving the hovering issue by manually bringing them onto the ground with keyframes and the final being the head position.

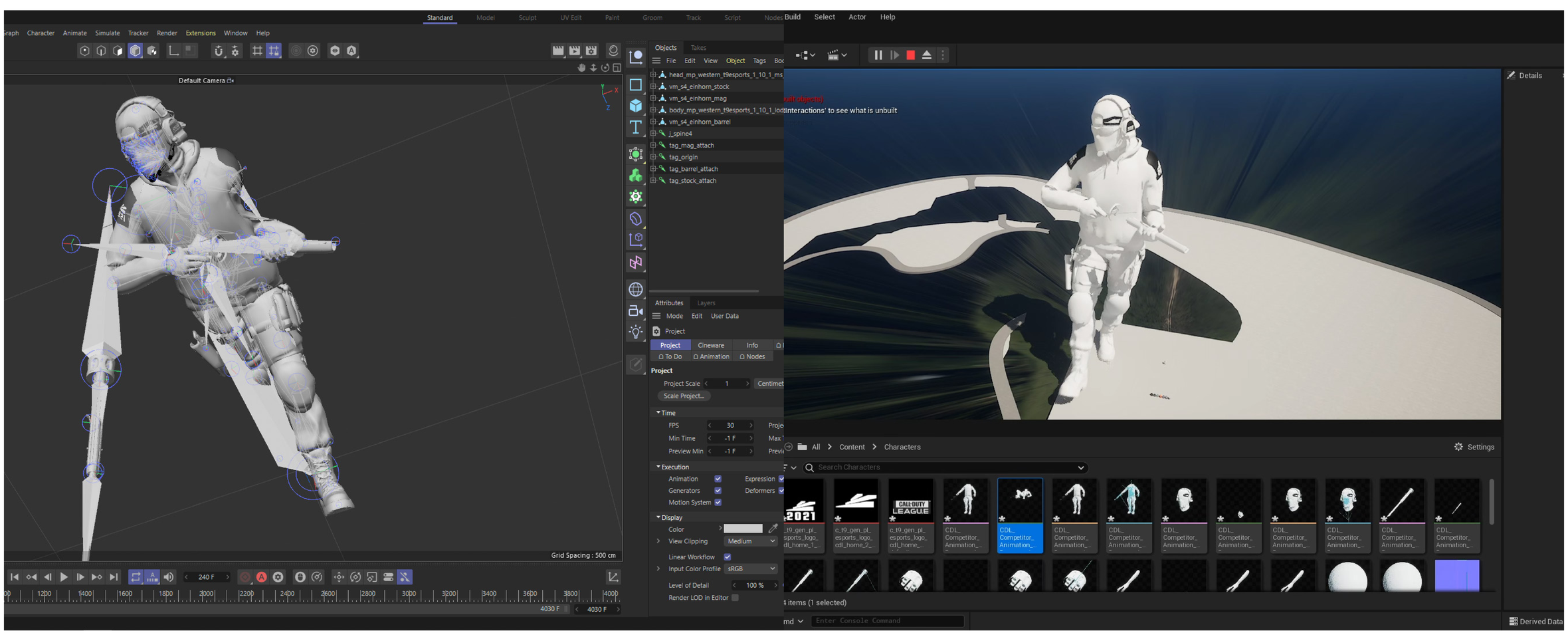


EXPORTING

The next big hurdle was definitely exporting given the flurry of issues I encountered that almost jeopardized the project had the eventual solution came to fruition. The first obvious initiative was exporting with the native FBX exporter in Blender, although it didn't bake the animations and it wasn't correct. Next was to bake the NLA animations manually with the bake action tool, but Blender would crash. Worried on what to do, I went with more advanced export formats like GLTF and USD neither of which baked the animations too and finally Alembic given the nature of exporting the animation as a per frame mesh snapshot type of system. This has proven to work when importing in other 3D applications like Cinema 4D, Maya and 3DS Max, although strangely Unreal Engine had the separate mesh parts located incorrectly even though it wasn't in the other applications. Before going into making the Alembic export work, I've perused for other solutions to exporting from Blender to an addon called Better FBX Importer & Exporter by Mesh Online that features a better algorithm when exporting to Unreal Engine/Unity.

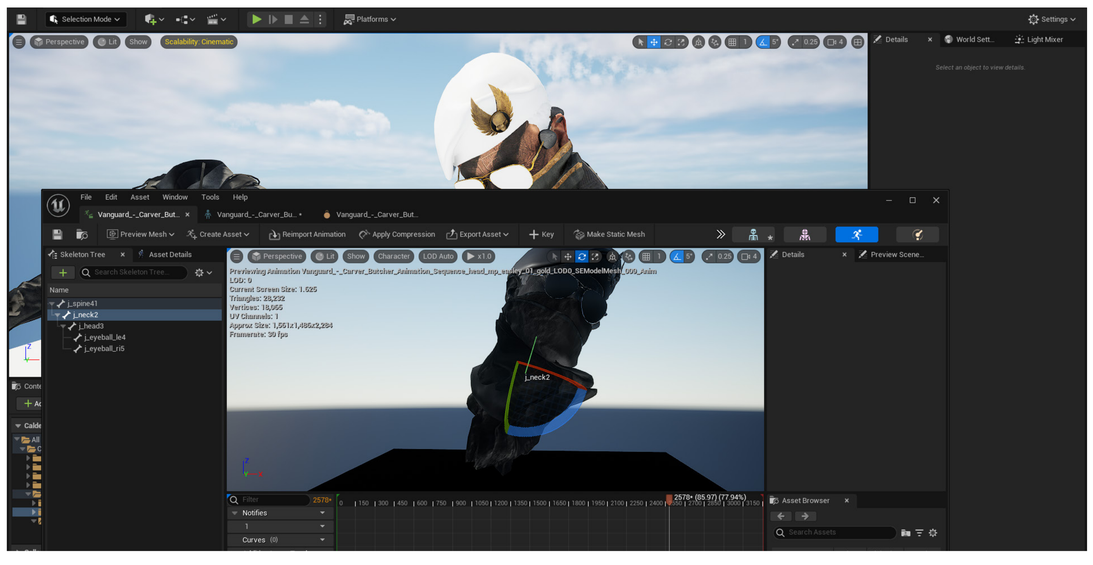


As such with nothing else to lose, I've tried the addons exporter with default settings and with joy found it correctly exporting in Unreal Engine with no rotation/location issues for the character. The weapon didn't export correctly but thankfully we can easily solve that by exporting the weapon separately and attaching it to the character by adding a weapon socket and attaching to that like how I did with Blender.



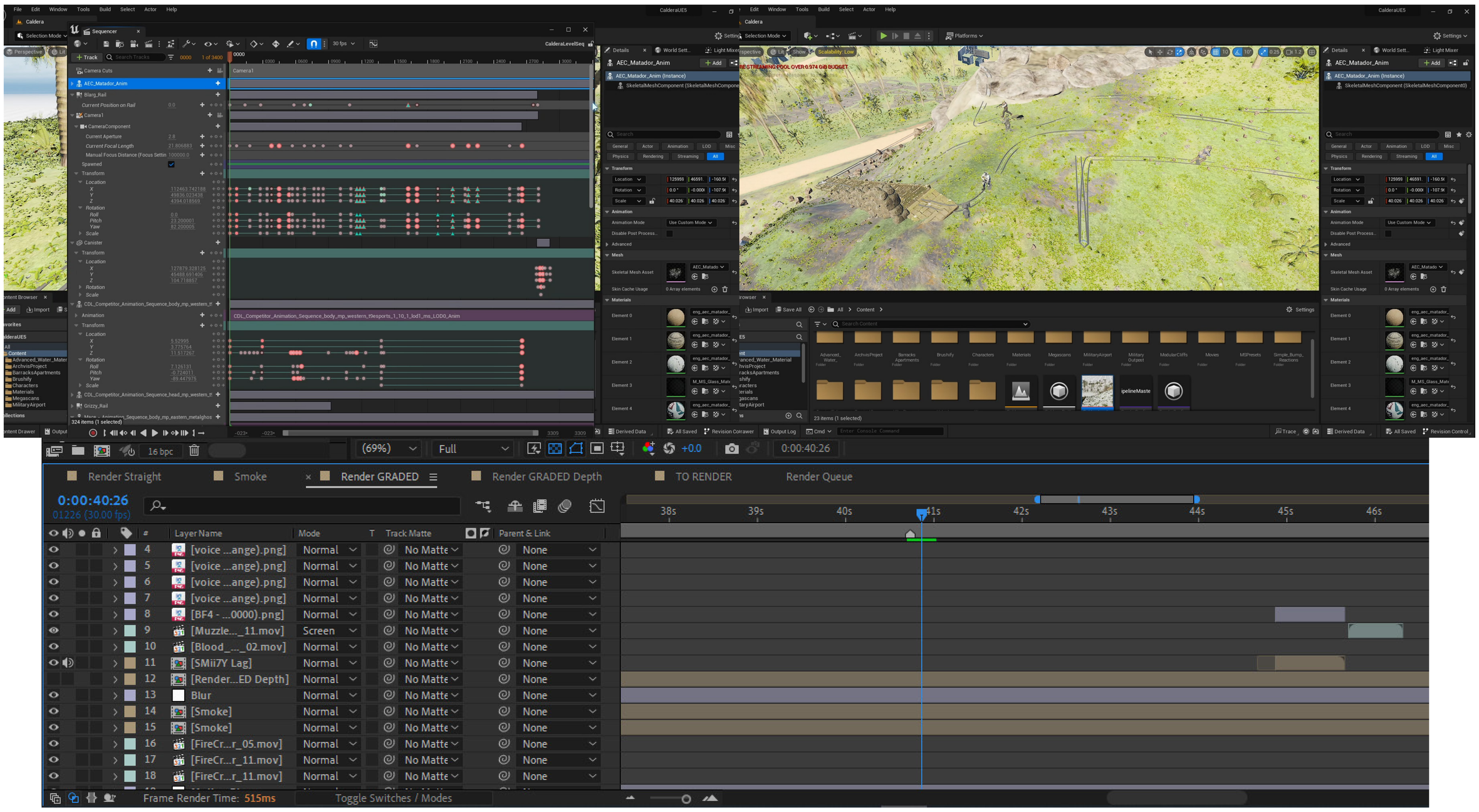
ANIMATIONS AND SEQUENCER

Once all animations were done with custom animation layer adjustments for each character, all animations were imported to Unreal Engine correctly with one exception of Carver Butcher, the character with the Nerf gun, that had his head rotation 90 degrees to his right for some reason even though it didn't happen to any other character. For this I simply added an additive layer track with his head adjusted correctly in Unreal Engine.



Timing wasn't required given all of the characters were 3309 frames at 30FPS, so it was just a matter of dragging into the sequencer and letting it play as such. The next hurdle was having them move about in the scene given the obvious decision to have them in-place, for this I went with adding them to a path that has a start and end point that's animated along the time, as such the camera rig railway tool was useful for this. Adding the path was a matter of tracing roughly to the landscape and setting end points. Once done for all 4 characters, their completion of the path was timed to the sound composition alongside more tedious work on adjusting the rotation and location to look like they're snapped to the ground such as if they're going down the hill.

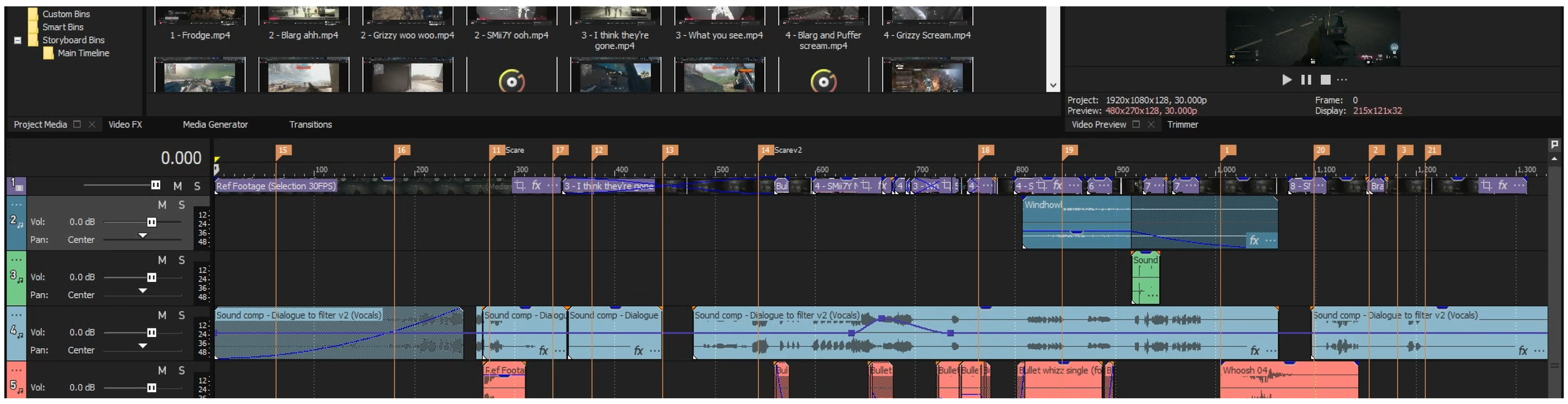
Once all done, it was a matter of exporting the sequence as an image sequence for post-processing, mostly just adjusting the gamma and lumetri and adding simple FX like smoke ploofs and firework bursts.



FILMOGRAPHY

The sequence and timings of when each scene happens and what camera angle was very straight forward given how determined I was based on what I had planned out months ago, the only scene that I was fumbling around for what exactly was going to happen was when SMii7Y died and Puffer starts worrying. This was due to how I originally wanted the camera to lock onto his face as he was backing up and it didn't cut between his back and him dying to the Carver Butcher. Although due to lack of facial capture that I was anticipating, I had to resort to a camera angle that looked from below, as such the typical foot shot of someone carefully going back was good enough followed by a brief back shot revealing the Carver Butcher emerging from behind the bushes.

For the beginning shot being one take, I decided on that due to how I wanted the chaos to ensue in full force between the gang so that it doesn't cutoff and focus on one like as if it was an overview shot from a real time strategy game seeing all your characters. Which lend itself to panning around dramatically to end points between a low angle shot of Grizzy looking out for the sniper and swishing to SMii7Y before doing the classic back and forth camera focus between the two going head to head. I didn't want to overstate though doing multiple back and forth to keep it fast paced and not drawn out given it was already going to do that when slowly transitioning between Puffer's demise and Blarg in the tent.

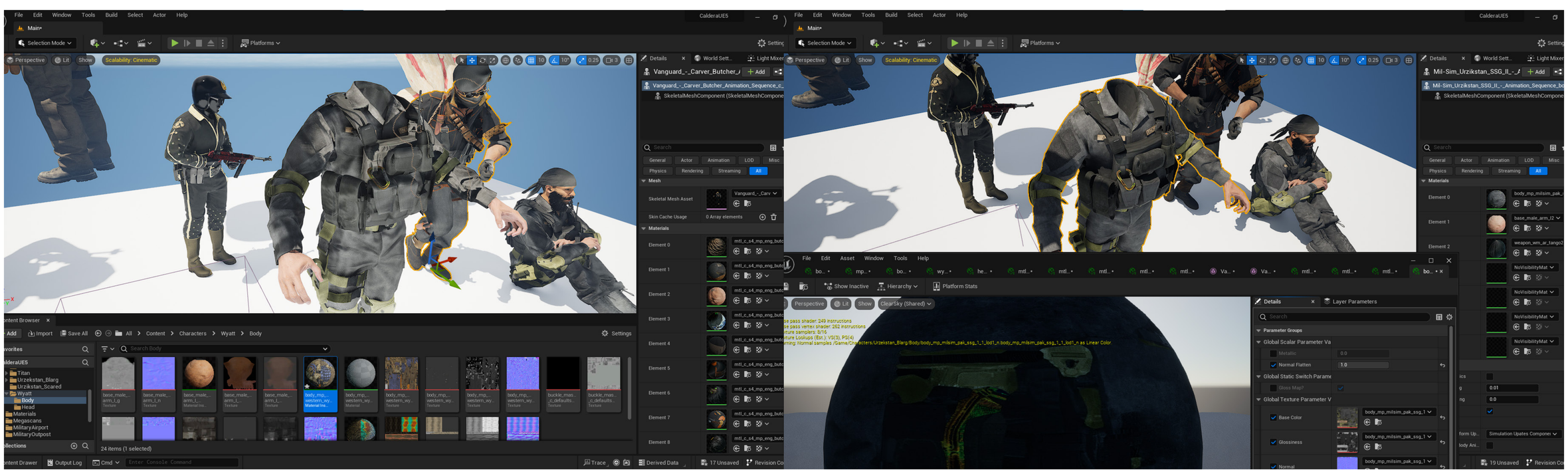


The sequence being dictated by the dialogue and sounds admittedly did lead to some pacing issues as it feels like the transitions between each scene such as when Blarg shoots the rocket and runs away to Grizzy going up the cliffside went too fast and didn't leave enough breathing room to let the viewers know the camera focus is now on Grizzy. This boiled down to not wanting a lot of dead space between each of dialogue, but inadvertently made the camera catch up. As such I should've sourced more dialogue to fill in between spaces given there's a few duplicates used already.

OTHER TECHNICAL HINDERANCES

One of the first leading technical decisions that hindered the project was rendering the height map at 8K that in turn made the landscaping tool already lag compared to using a more typical 4K resolution. This subsequently made actions like using the landscaping brush or foliage tool more tedious as it would have to take time to process in between. This was discovered halfway through landscaping so I stuck to utilizing the 8K height map instead of rendering the height map at 4K.

The last one I resolved, under the notion that the weird lighting appearance was only temporary as I had set the engine settings to low whilst working on the project, was the normal map issue with the Call of Duty characters and weapon models. Even though this wasn't present for the meshes that were exported from Blender without a weighted normal modifier that were sourced not from Call of Duty, it was apparent on the Call of Duty models as applying the normal map presented the weird shadow/shading issue given the interpolation of the normal axis was being distorted. Even after trying to resolve it by both flipping the y-axis in the textures properties and multiplying the normal vector by (1, -1, 1) to undistort the normal, it didn't avail anything.



https://www.reddit.com/r/unrealengine/comments/v9g88y/normal_map_issues_ue5/ic02gkg/

After double checking it wasn't the model's normal tangents by just using another generic material, I decided on flattening the normal or in other words lowering the strength to 0 as a quick solution given that the normal maps kind of don't matter for the most part given the mostly lack of close-up shots that'll benefit with a normal map for the character.

CONCLUSION

Using animation packs only were as expected undesired in some places, although handy in other situations, mostly in the scenario of how the modularity has been done with the different speed and direction choices being provided helps with the workflow for scenarios as with the initial shot scene. Although motion capturing custom story specific actions like when SMii7Y confronts the Armored Titan to Blarg hiding in the tent would go the mile long to convey a more exaggerated look to the scenario at hand given the current look gives off a video game Machinima look with just the video game's animation playing with no taunts or expressions. Hopefully motion capture could've also fixed the hurdle of having to adjust the animation manually to hold the gun from their left hand correctly instead of offsetting continuously. For the traversing of the characters in-place, the current solution of using paths works best only in Previz terms, whilst a more proper way is something I couldn't find about, most likely being a case through advanced locomotion rigs or more continuous motion capture sessions without blending to eliminate that need in the sequencer/NLA editor. The other obvious additional work would've been facial capture that would've required working with the system at hand and retargeting that onto the face bones too with a flurry of potential issues given the inexperience.

Lighting wise, Lumens is pretty good, but obviously not as self-sufficient as Blender Cycles that simply generates photorealistic renders with a simple HDRI and accompanying directional light setup compared to Unreal Engine. Granted there are benefits to Unreal Engine being the real time performance needed to see a dynamic sky, windy foliage and the game changer of Nanite compatibility with foliage that allows for expansive nature sceneries that Blender would heavily suffer when processing that much geometry. I'd reckon learning more on how to make my scene look less bright and natural would heavily influence the final look given my lack of knowing how to tackle lighting situations compared to Blender that requires little to none focus on that aspect.

Exporting as discussed before was a potential nightmare and game breaker given the nature of utilizing bone constraints and NLA blended animation clips and animation layers that's more than the expected exporting scenario than usual. Had the render been in Blender it would've obviously not needed to go through a phase of translating the work to Unreal Engine. This also applies to the materials since the Call of Duty materials required a little bit more intricate node setup including blending between the specular and color map with a luma matte and the normal map issue that wasn't working out in Unreal Engine whilst Blender correctly showed it without any additional nodes needed. Additionally hair cards didn't render out as proficiently as Blender given the nature of anti-aliasing far away objects.

Finally the aspect of the other scenarios proposed in the beginning including the bit where their chibi counterparts comes to life and play out with the characters, the adventure to the top of Peak and going through the wormhole back to Verdansk comes down to time constraints and the need to go thoroughly over Houdini for the storm simulation which would've required going through VDB implementation and using the path tracer in tandem with Lumens.

CREDITS

Characters:
CDL Competitor - Call of Duty: Black Ops Cold War (Treyarch)
Mil-Sim Urzikstan SSG II, Mace Executive Armorer and Wyatt Outback - Call of Duty: Modern Warfare 2019 (Infinity Ward)
Carver Butcher Gold, T-1000 Gold, Night Terror (Roze 2.0) and Armored Titan - Sledgehammer Games

Animations:
Rifle Pro - MoCap Pack by MoCap Online @Unreal Engine Marketplace
Simple Bump Reactions - RamsterZ @Unreal Engine Marketplace

Weapons:
AS-44, Bazooka, Becker Revolving Shotgun, Kar98K, Trench Shotgun, MP40 - Sleghehammer Games
Nerf Stampede by Me

Vehicles:
GMC CCKW-352 - Call of Duty: WWII (Sledgehammer Games)
AEC Matador - Call of Duty: WWII (Sledgehammer Games)
Vought F4U Corsair - Panaristi @Sketchfab
Crane with wrecking ball - krzysztofgradowsky @Sketchfab
Military Jeep - soidev @Sketchfab

Scenery/Environment:
Tropical Foliage and Landscape - Dekogon Studios @Unreal Engine Marketplace
Military Airport - FreshCan @Unreal Engine Marketplace
Military Outpost - Kitbash3D
Modular Cliffs - CGMontreal @Unreal Engine Marketplace
Brushify - Cliffs Pack
LKPR_Military_Terminal Prague Ruzyne Airport by round @Sketchfab
Military Base Barracks Buildings - CommonSpence @Sketchfab
Beach Boulder, Forest Rock, Japanese Shrine Stairs and Mine Cart from Quixel Megascans
Tropical Ocean Tool - James Stone @Unreal Engine Marketplace

Props:
Barrels, Drums & Containers - Low Poly - MSWoodvine @Sketchfab