Mocap Effects Setup & Explanation

Effects in the scene:

- Lightsaber with trail
- Axe with Lightning
- Baseball bat with chaos fields (emits fire at certain velocity)
- Ball launcher with destructible baseballs
- Destructible wall
- Interactive water (with correct props)

Unused effects (potentially can be used with mocap suit):

- Flame body
- Footstep dust
- Portal
- Lightning field

Scene layout:

- 4 gold cubes mark general bounds relative to the motion capture lab
- Ball launcher is stationed over brown platform

How to create a tracked object in Unreal*:
*Assuming your Vicon setup is calibrated and already has a defined prop

- Make sure your project has the ***Live Link*** and ***Vicon DataStream LiveLink*** plugins (Live Link can be found on the plugin search bar, Vicon plugin has to be added manually)
- At the top of the screen, navigate to ***Window > Virtual Production > Live Link***
- At the top left of the Live Link window, click the ***Source*** option with the green plus icon. Move to ***Vicon Data Stream Source***. Assuming the Vicon setup is correct, leave all the fields default and click “create.” You should see the name of your Vicon tracked prop in the lower list now.
- Add an object into your Unreal scene (Easiest to adjust orientation if it’s a stick or something)
- Add the ***Live Link Controller*** component to your object.
- Select the controller component and look in the details panel. Click the ***Subject Representation*** dropdown, and select the name of your Vicon tracked prop.
- If your object disappeared, it probably got teleported to the origin of your scene.

- You should now be able to see your object moving around in the scene as it moves in reality.

- If your object is not moving/facing the right way, go back to the ***Window > Virtual Production > Live Link*** window. Click on your tracked prop on the lower list.
- On the right side of the window, click the plus button on the ***Pre Processors*** section. At the Index[0] dropdown, select ***Transform Axis Switch***.
- Click the dropdown arrow on the left side of the Transform axis switch section, then again on the Live Link section. Enable the ***Offset Orientation*** checkbox, and change the Y-axis field to 270*.
  *This works with the Vicon setup as of 18/08/2023. If it changes in the future this may need to be tweaked.

Water:

- Create a new ***Niagara System*** asset
- When prompted, select “***New System from a template or behaviour example***”
- Scroll down to the 3D water section and select ***Grid 3D FLIP Pool***
- Leave all settings as default, drag your pool into your scene
- Select your pool and scroll down in the details window to the ***User Parameters*** section. Change ***World Grid Extents*** and ***Water Height*** if desired.
- Select your tracked object or any object you wish to collide with the water. Add the ***Actor Tag*** “collider” to it.
- Your object should now be able to push the water around. The water is pretty buggy and may not react properly or at all on the first try. Use the reset button on the pool and enter and exit play mode multiple times until it works.

Destructible objects:

- By default, the baseball bat is able to destroy any chaos object (wall, baseballs).
- To make an object able to destroy other objects, you must attach a **FS_MasterField** to the object you are using.

- The baseball launcher is on by default, and shoots baseballs towards the motion capture area. It is set up to shoot baseballs in that specific direction, so if it is moved then it will have to have its shooting direction changed in its blueprint. To stop the baseballs from shooting, you can either move the shooter out of the way or delete it. When you add it back, it will have the correct orientation by default, just make sure to put it back over the brown platform.
- To move the wall into the scene, make sure you move both **WallDestrucible** and **WallAnchor** together.

    - The anchor is used to make sure the entire wall doesn’t fall over after getting hit

Creating a destructible object:

- To create an object to be destroyed by another we can do so with basic shapes in the editor, or any object imported into the scene (some won’t work as well depending on the complexity of edges).
- As an example we can take a basic cube from the editor.
- After you’ve dragged and placed a default cube into the scene, select the cube, then go to the top left of the window and press the dropdown menu where it says **Selection Mode**, and change that to **Fracture Mode**.
- Once you’re in **Fracture Mode**, under **Generate** you’ll need to press **New**, select any designated folder for this new collection to be generated in (This case, just use the chaos folder), name it whatever you’d like (default name is fine), and hit **Generate Geometry Collection**.
- From there, scroll down on the left hand side menu to the **Fracture** section, and select **Uniform**.
- In this selection there’s a lot of advanced settings you could play with, but the main thing we’ll be using is **Uniform Voronoi**.
- Under **Uniform Voronoi** change the **Min Voronoi Sites**, and **Max Voronoi Sites**, to either a range on min and max or the same value on both min and max to generate the amount of fractures in an object.
- Generally more is better for more fragments, so inputting 200-300 for min, and 200-300 for max is solid (I input 300 on both min and max)
- Once both min and max **Voronoi** is set, scroll down and in that same window until you see **Fracture** and press that.
- Now you can swap **Fracture Mode** back to **Selection Mode**, select the cube and go to **Show Bone Colors** and tick that to false if you’d prefer to not see the fractured lines in render view.
- This is my personal tip, but if you type in the search or find **Damage Propagation**, tick **Enabled** as false, and the objects destroyed will fragment much better.
- One last thing to stop the Object from destroying itself if it’s floating even by a pixel, is to drag and drop from the **Chaos** folder in the **Content Browser**, is **FS_AnchorField_Generic**.
- When the **FS_AnchorField_Generic** is placed into the scene, make sure it’s placed where you’d like to anchor the object and have it overlap a tiny bit on the object itself, keeping it in place.
- Then finally, select the Object you’ve made destructible, search or go to **Initialization Fields** under **Chaos Physics** and press **Add Element**.

- You can select from the hierarchy the **FS_AnchorField_Generic** you've just placed into the scene, or select the eyedrop icon, to select from the scene the **FS_AnchorField_Generic** and finally finish creating a destructible cube.
- Make sure Simulate Physics is enabled

Enabling a prop to destroy an object:

- To do this, we'll use **BP_Bat** as an example.
- Double click **BP_Bat** and go to the viewport. Once there, make sure nothing is selected and then drag and drop the **FS_MasterField** into the hierarchy from the **Chaos** folder in the **Content Browser** and move in the viewport to the desired location.
- You can shrink or enlarge the **FS_MasterField** to suit the use case (Note that whatever touches this sphere will be hit with an impulse).
- From there, make sure the **Activation Type** under **Child Actor Component** is changed to **OnTick**, as it is default to Delay, which activates the field only once.
- Furthermore, make sure under the Child **Actor Component**, the **Use Tick** node is set to true, as without it the field will update once and never again.

Using the BP_Enviro_Lightning asset:

- Currently the BP_Axe asset has a BP_Enviro_Lightning as a component. The way it works is by pointing the axe at a tagged object, a particle system will be created which arcs lightning from the head of the axe to the tagged object.
- In order to create targets for the axe, give the target object the ***Actor Tag*** "target" and ensure it has the "***Generate Overlap Events***" setting set to true under the collision header. The default collision preset should work, but if it doesn't, set the collision preset to "overlap all"

- To create new objects that can fire lightning, add the BP_Enviro_Lightning object as a child actor or as a component in a blueprint. By default, the BP_Enviro_Lightning components are invisible.
- For testing purposes, you can change the BP_Enviro_Lightning "root" and "collider" components to be visible by looking under the "rendering" header of their details window and toggling "visible." Do not attempt to make the whole BP_Enviro_Lightning child component invisible as this will also make the lightning invisible; change the visibility of the "root" and "collider" objects only.