UE4 C++ CODE PLUGIN – DISTANCE MATCHING

by Berkay Tuna

Table of Contents

- Introduction 3
  - What is Distance Matching, Why Do I Need It? 3
  - How Does the Distance Matching function? 3
- Implementation 4
  - How Do I Set It Up? 4
  - Creating the Curve 11
    - *AnimationModifier_DistanceCurve* 11
    - *AnimationModifier_CopyCurves* 13
    - *AnimationModifier_ChangeCurve* 14
- Troubleshooting 15
- Summary 16

Introduction

- **What is Distance Matching, Why Do I Need It?**

Distance Matching is a part of locomotion system. Together with Animation Warping and many more innovative features it provides a realistic movement for characters. Originally used in game Paragon from Epic Games and its animation features were presented by Laurent Delayen, their Lead Animation Programmer. Please check his Youtube Channel for the presentation and to learn more about the Paragon Animation Features.

Users of Unreal Engine have access to many interesting Animation Packs, thanks to its Marketplace, but a system that makes full use of these Animation Packs is hard to come up with. Distance Matching is aimed at using Start, Stop and Rotation animations efficiently and realistically. Original Distance Matching application covers Pivot Animations, as well. **However, my plugin does not include Pivot applications!** Instead I chose to make use of Blending between States (which in my opinion functions very good), as most of the Marketplace Animation Sets do not include Pivot Animations.

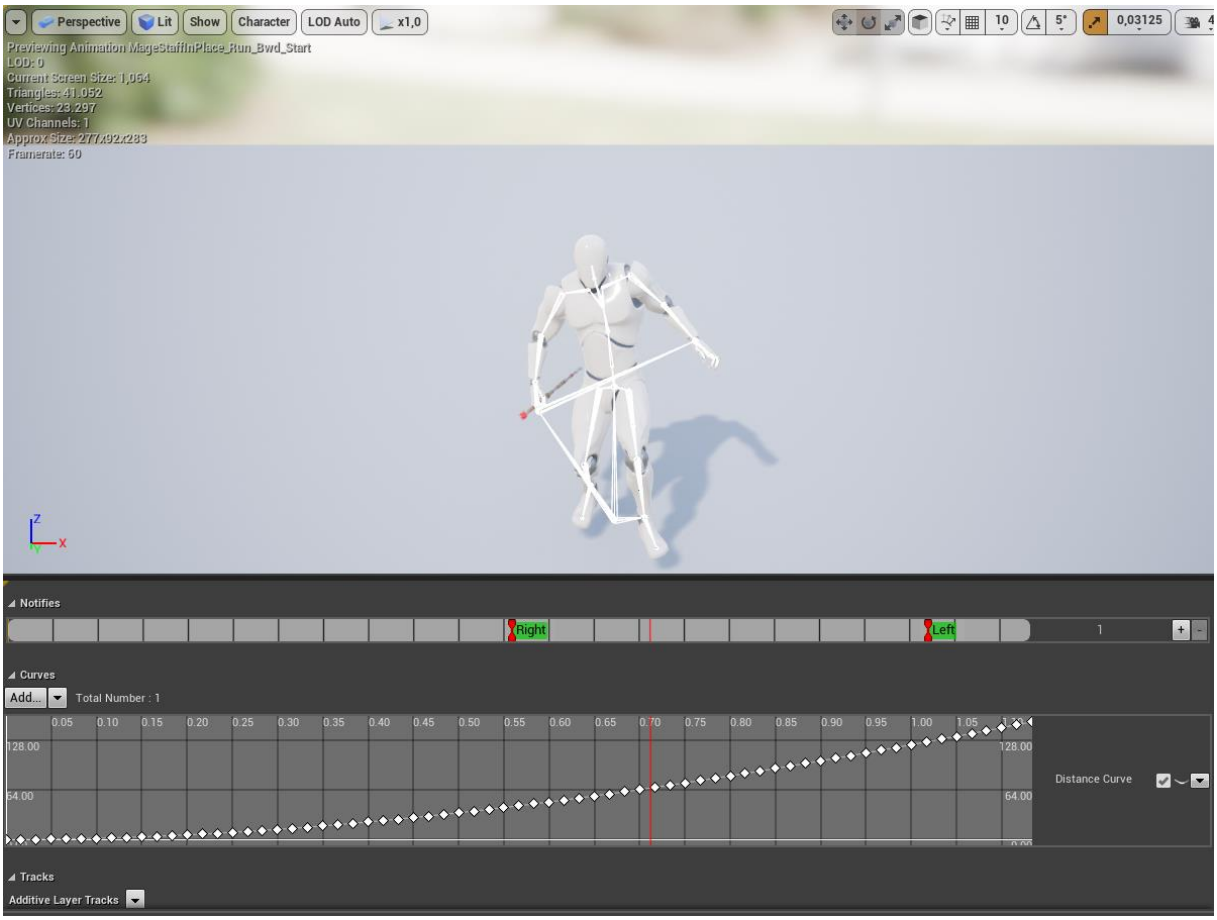
- **How Does the Distance Matching Function?**

The system will place a marker in the world for the Start, Stop and Rotation positions. When the Actor gets closes or gets away from these positions, as a result we get a distance every frame. Every Start, Stop and Rotation **Root Motion Animation** can be used to get the according curve for its Root Motion. Therefore we have the access to the distance to the Calculation Point for every frame. The Distance Curve has Time in Y-Axis and Distance in X-Axis. When we provide the distance we have calculated to the curve, we can then get its Time counterpart through C++ methods- Doing this every frame during the Animation playing, we know at which frame at what time Animation should be at. When we provide this information to the single frame Animation Sequence in the AnimationGraph, we can then control the Animation. Therefore it will always match our Character`s speed and it will look realistic.

Implementation

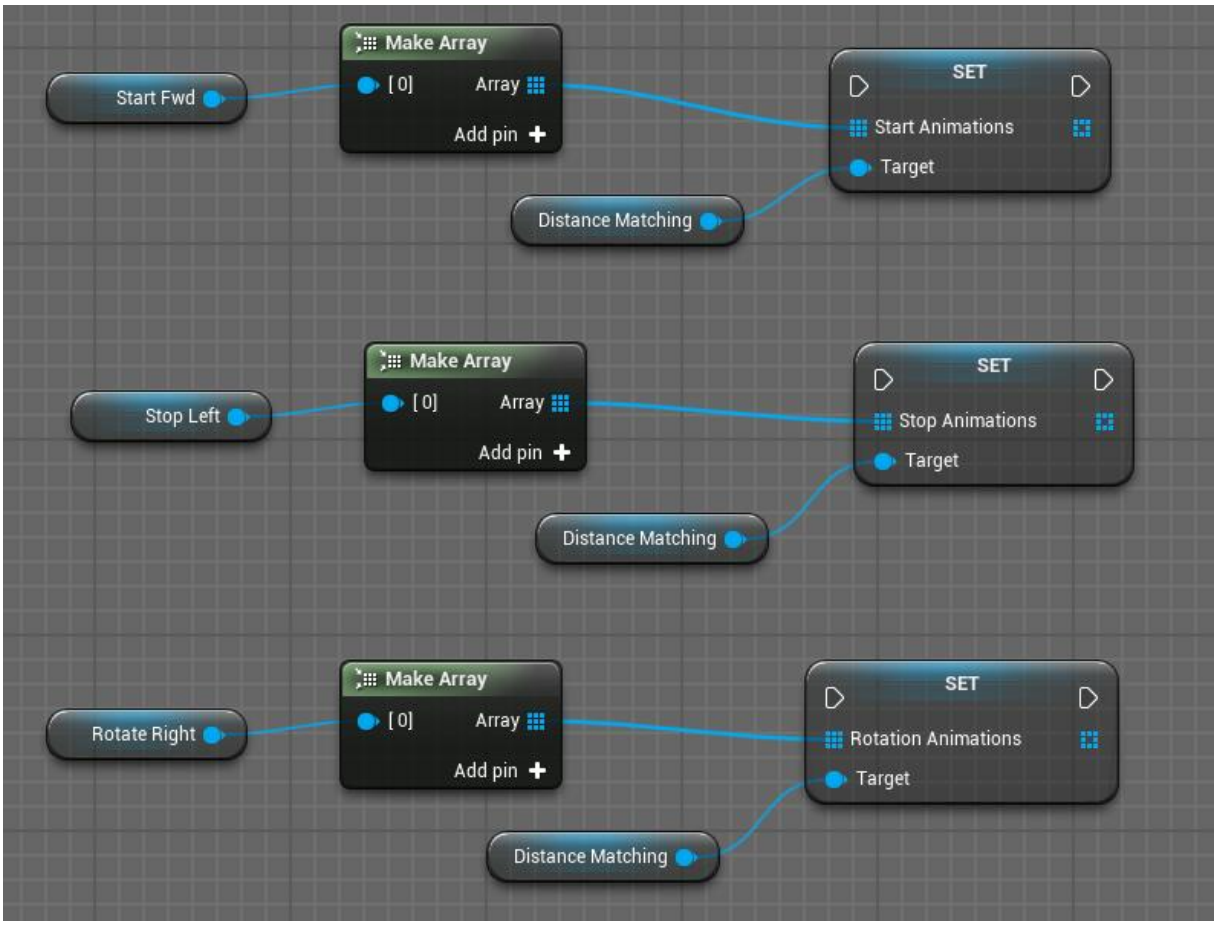
- **How Do I Set It Up?**

As mentioned before, we will begin with our curve. I will get into more detail on how to obtain these curves in the next chapter.



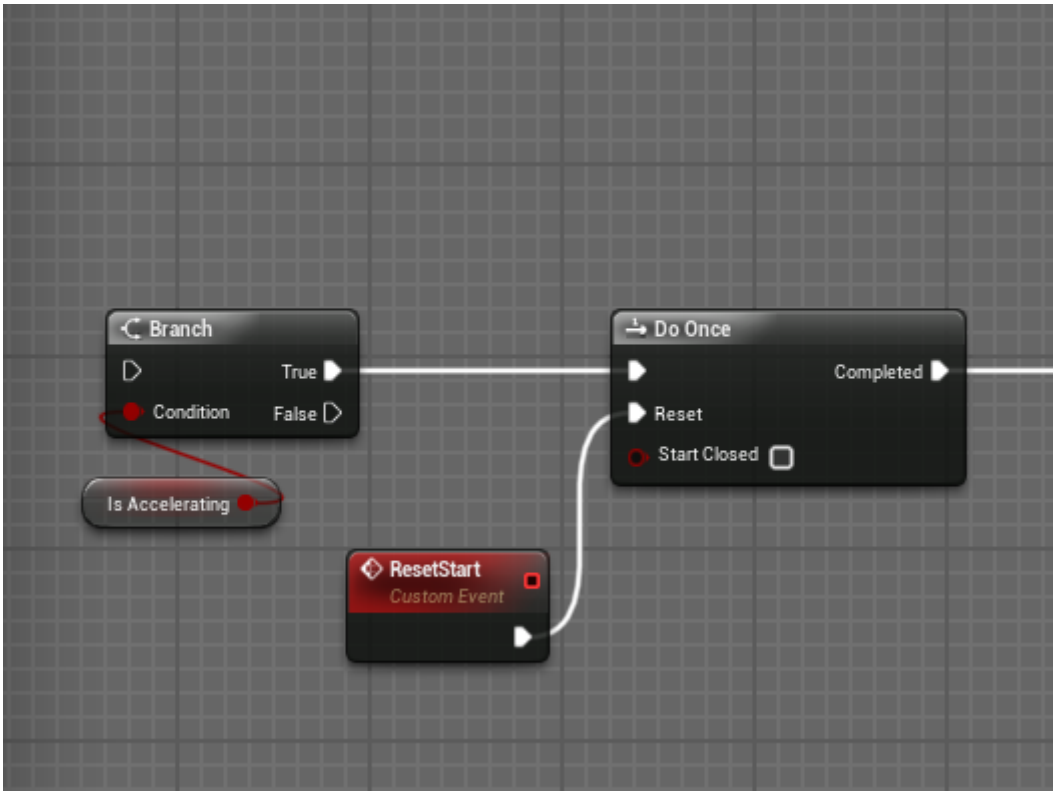
Picture 1: Distance Curve of a Start Animation

This plugin contains three main Arrays, which can be set by user. These are Start, Stop and Rotation:



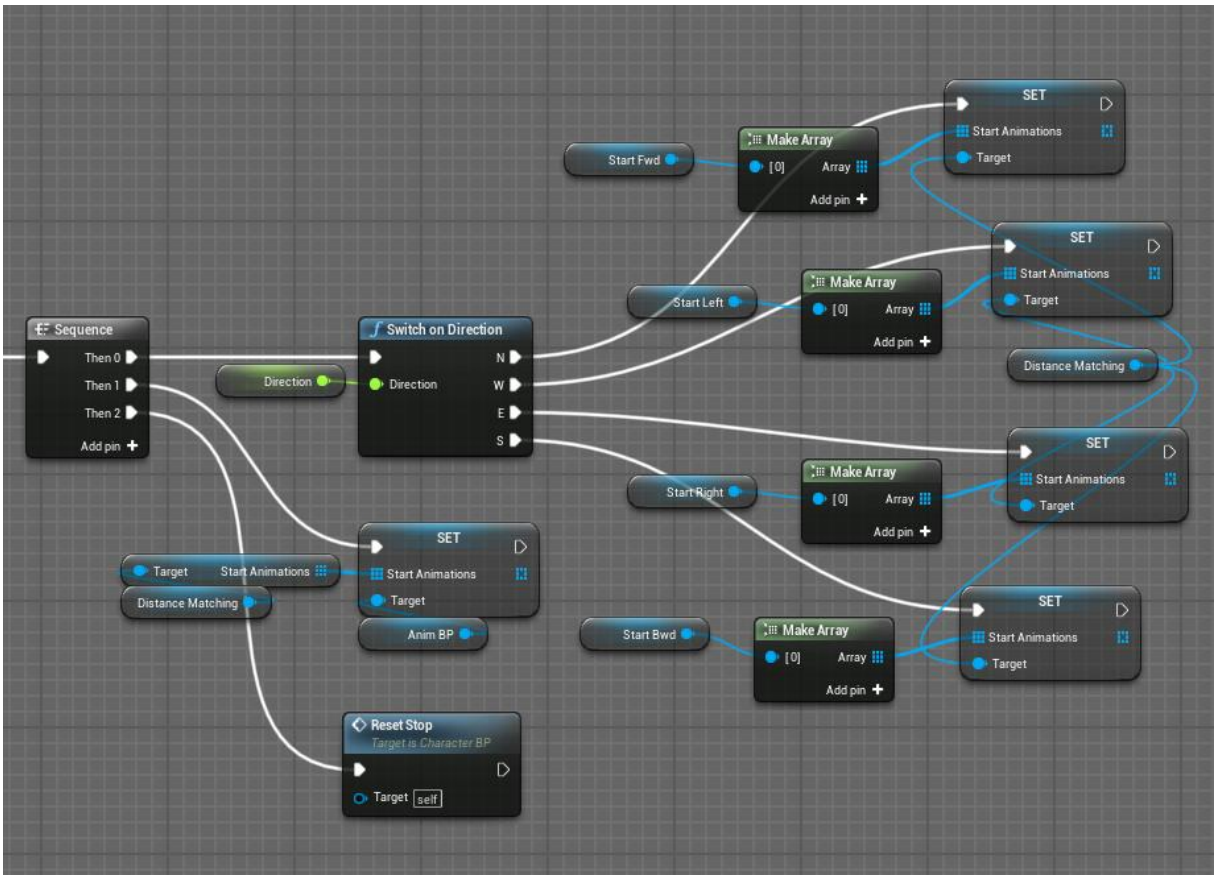
Picture 2: Start, Stop and Rotation Arrays from the Distance Matching Component

When we want to let the Actor Component know that it should begin to calculate for a certain Animation, we set the respective array with the Animation we want to calculate for. Multiple Animations can be set at once, as it is an array. Setting this array will give component the Animation but the component will remove the Array next frame but save the Animation Sequence in an internal variable. **So, the Animation Array from the component is only to be used one frame and to set it.** Whenever it is set, the system will begin to calculate again. This means **DoOnce** Set it at the beginning of the Transition. Here is an example setup from my Example Project, which can be found in the description page of this plugin:



Picture 3: First part of setting the arrays parameters for Distance Matching

We begin with checking if the character is accelerating. We will use a similar approach when setting Stop Animations, but we will initiate when bIsAccelerating is false. If the character accelerates we use a DoOnce node to make sure we are setting the Array only once, otherwise the system will start the calculation from the beginning again and again. Next, we will see how are the Arrays set up:



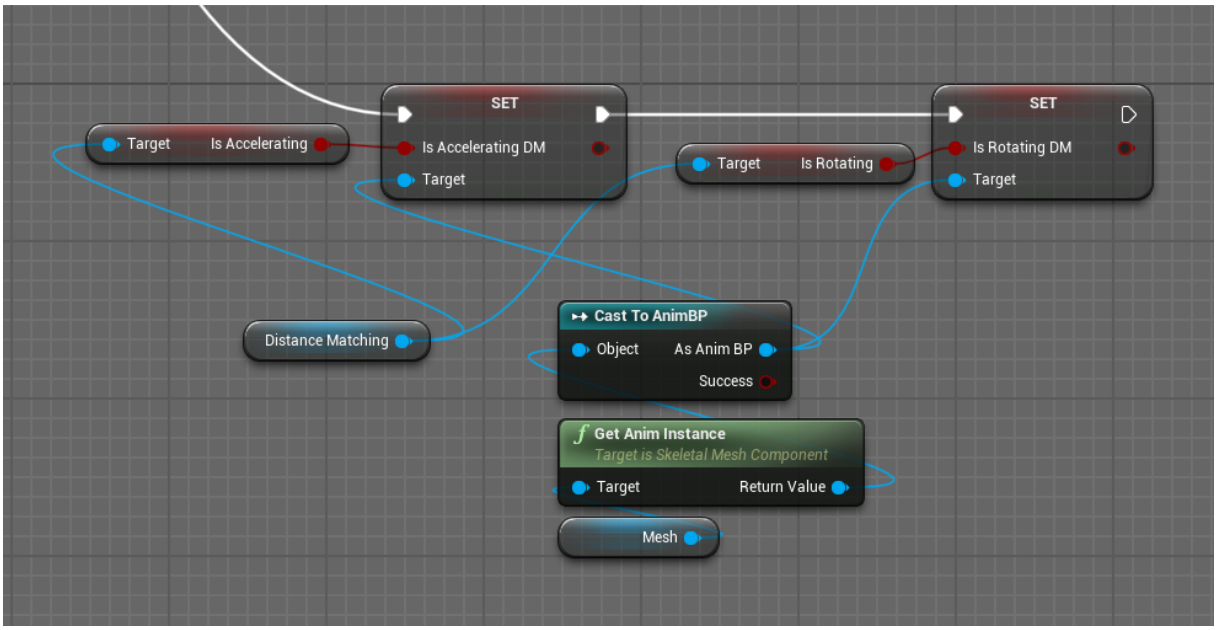
Picture 4: Setting Up the Start Arrays for Distance Matching Component

According to the Character`s Direction, we set one of the Arrays with the related Animation Sequence. In a sequence we set the Animation Blueprint variable **in the same frame**. And then we reset the Stop version of DoOnce. So, to sum it up, when we begin to accelerate, we set the Start Array of Distance Matching Component once with the Animation Sequence of the direction we are accelerating in. Then in the same frame, we set the Animation Blueprint Animation Sequence variable with the same Animation Sequence, as well.

Now our system knows which Animation we are going to use and let`s continue with the next part!

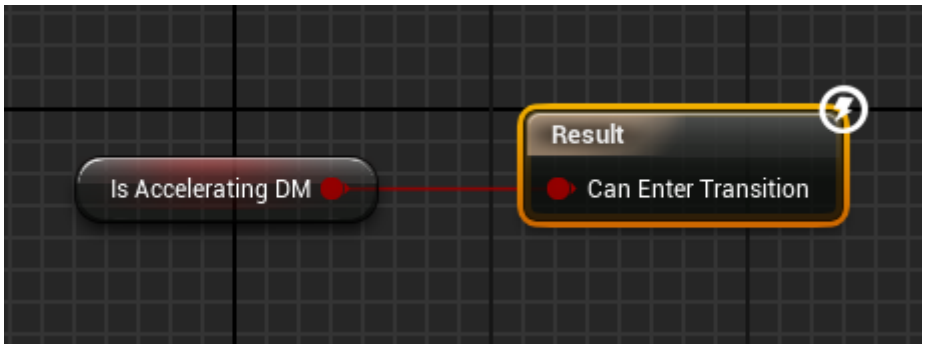
There is an execution order between Classes. This can be confusing. For example, The Distance Matching Component will begin to calculate after the Actor is completed with its calculations. To ensure that our system will always function correctly (which means Actor calculates first, tells the Distance Matching Component and the Animation Blueprint to use which animation, then Distance Matching component calculates and then Animation Blueprint

enters the correct state), we are going to use a boolean value from the Distance Matching Component. This is the Distance Matching Component saying “I got the Array, I know which Animations I have to calculate for and I initiated the calculations, you can now enter the next state”. The Boolean we are using is bIsAccelerating but it is coming from the Distance Matching Component this time:



Picture 5: Distance Matching Component`s bIsAccelerating and bIsRotating booleans and how they control the Flow in combination with AnimBP´s similarly named booleans in the whole system.

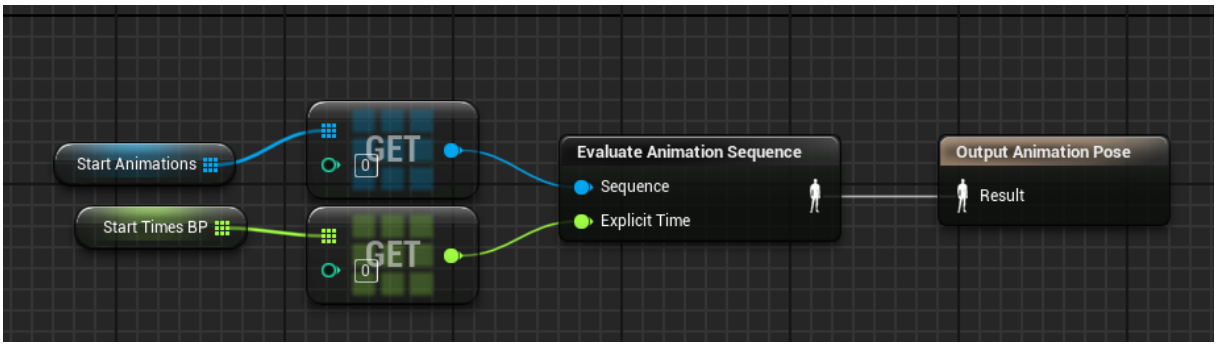
To understand this flow control better, let`s see how we enter to the Start State from the Idle State:



Picture 6: Idle to Start State Transition Rule

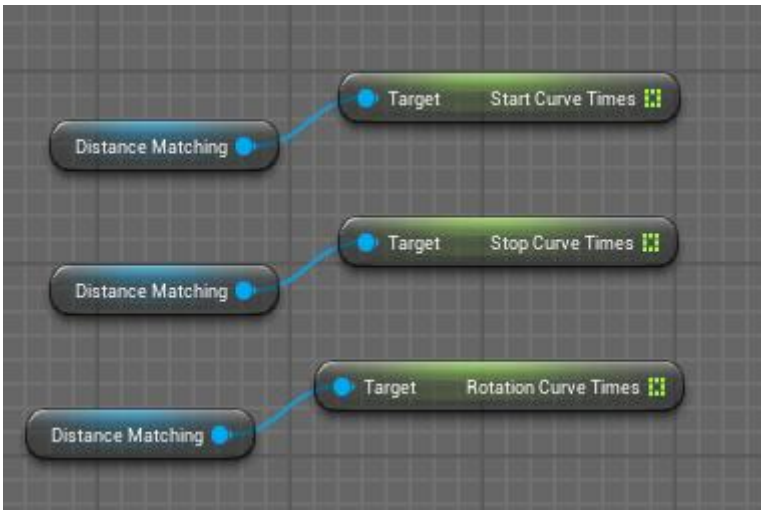
For a full setup and to see Stop and Rotation in detail, as well. Please download my Example Project from this plugin`s description page. It is free.

Now, we have the required Animation and we have begun the time calculation. Let`s see how Animation Blueprint uses this information in its AnimGraph:



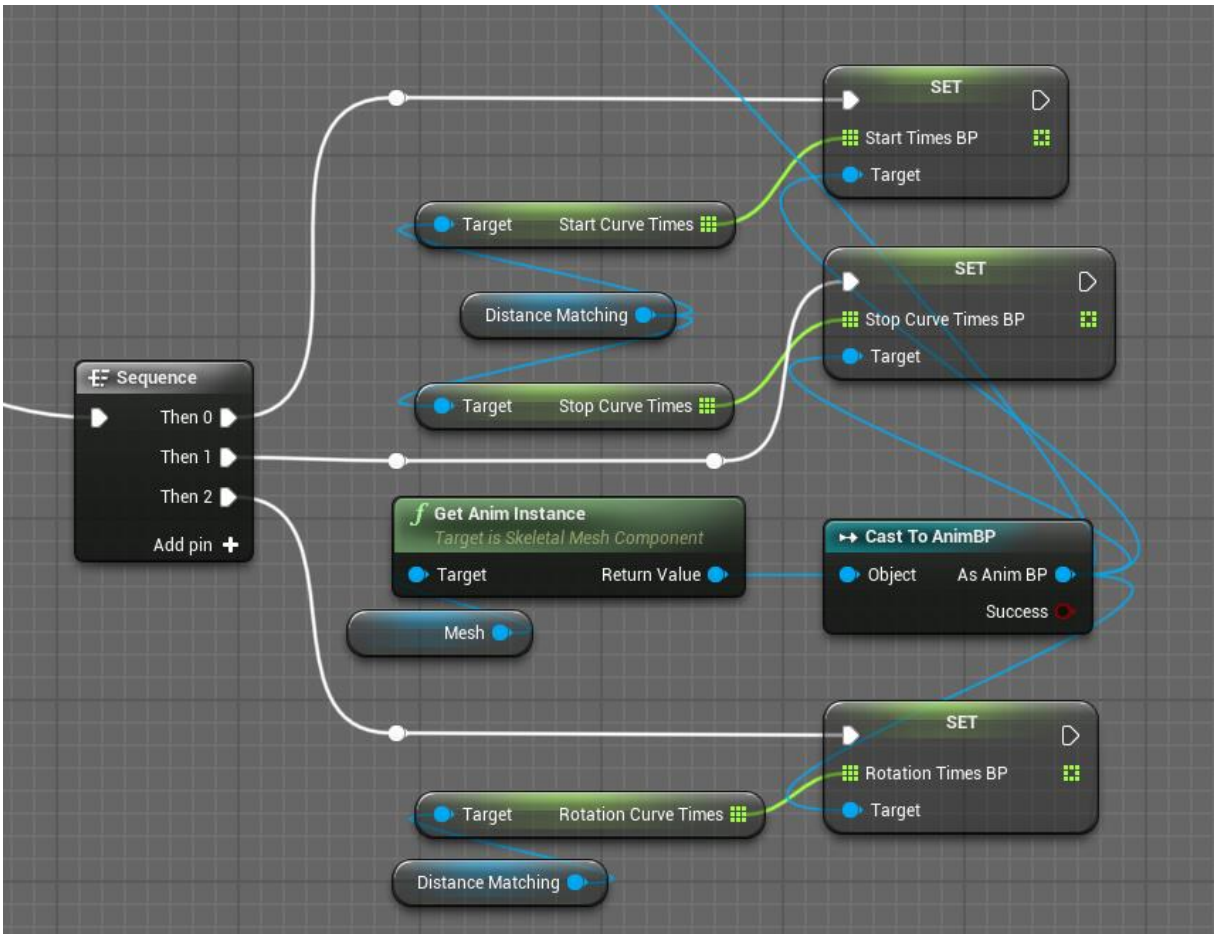
Picture 7: Inside of the Start State

The Distance Matching Component has a times Array variable for each state:



Picture 8: Times Arrays of the Distance Matching Component

These arrays give us the time value for us to use in the previously shown single frame animation application. In my example project I transfer the information to the Animation Blueprint like this:



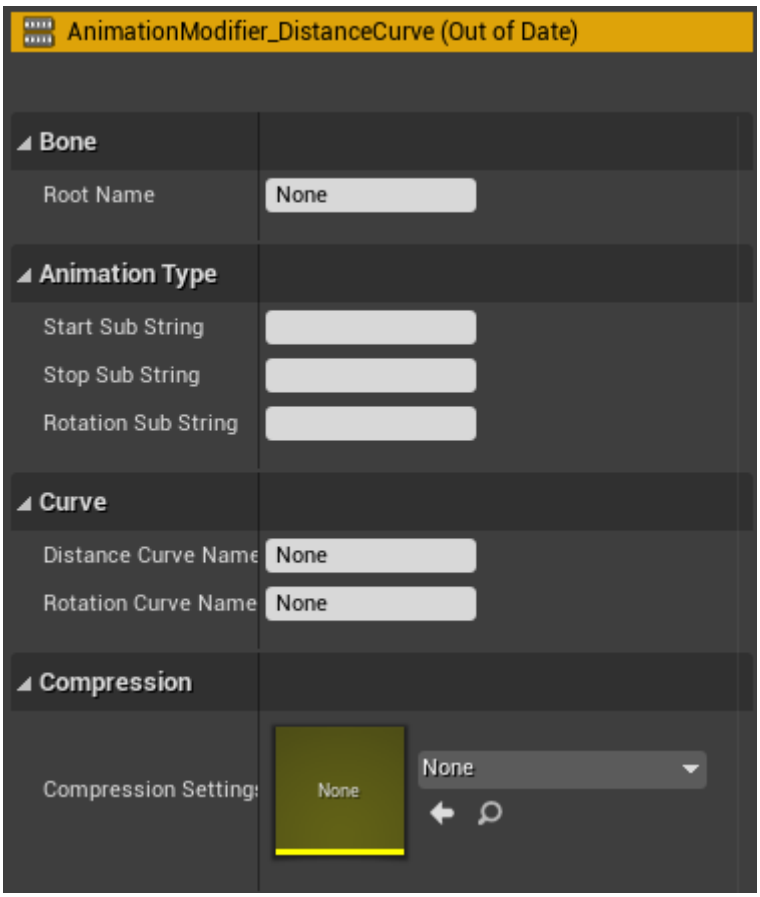
Picture 9: Transferring the time value from the Distance Matching Component to the Animation Blueprint

This is the logical setup of this system. For more, please check my Example Project. In the next chapter, we are going to discuss the curve creation process with this plugin`s included Animation Modifiers.

- **Creating the Curves**

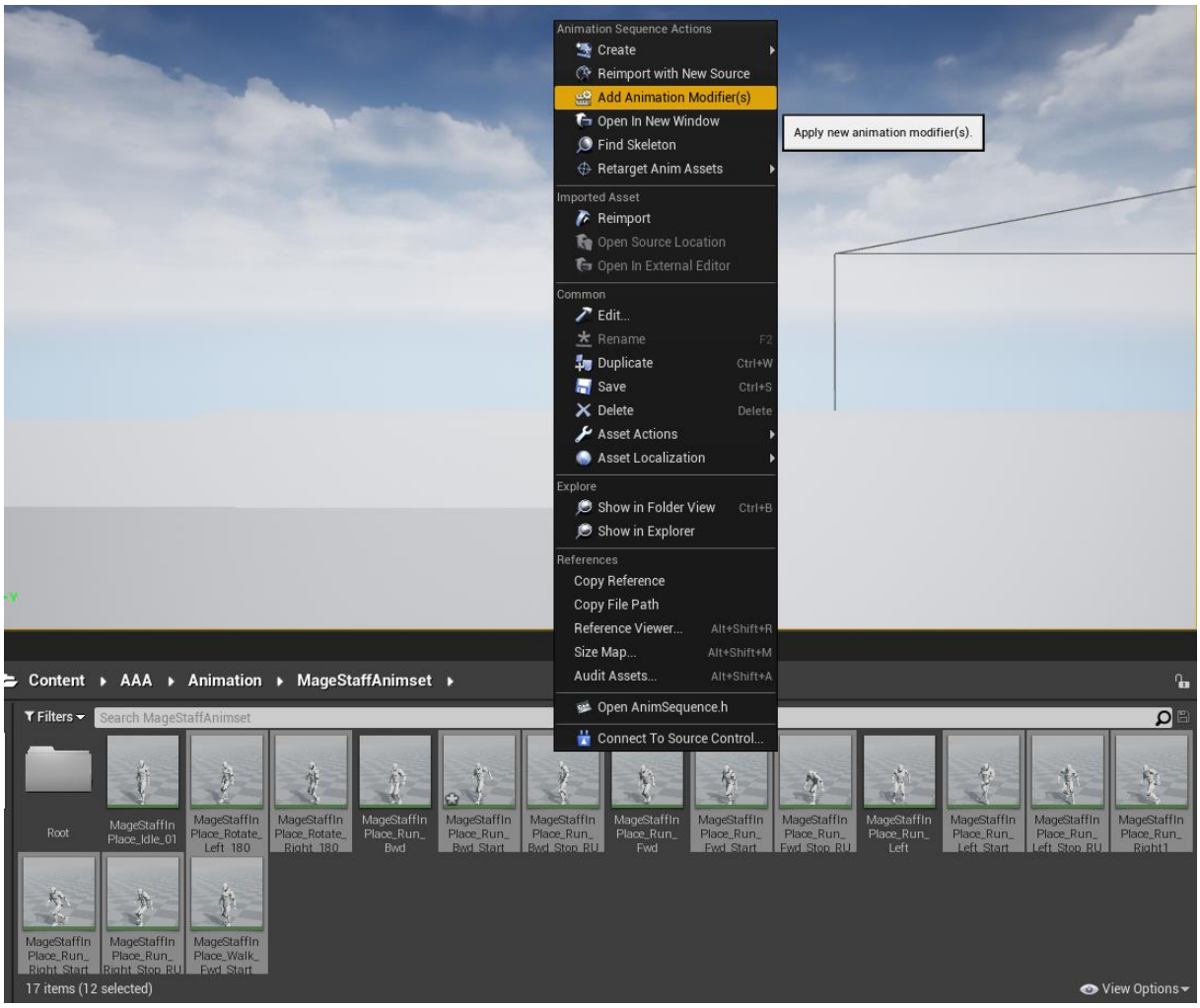
This plugin contains useful tools to help you create, copy or modify **multiple curves at the same time**.

1. *AnimationModifier_DistanceCurve*



Picture 10: AnimationModifier_DistanceCurve

We have 4 Start, 4 Stop and 2 Rotation **InPlace** Animations. We retargeted them from its original AnimSet folder to a new folder. In this folder, we created another folder named “Root”. Now, in this folder we have 10 **InPlace** Animation assets and 1 folder named “Root”. Similarly, we retarget 10 **RootMotion** Animation assets to the “Root” folder we created. Now in the folder we created, we have 10 **InPlace** Animation assets and 1 “Root” folder. And in this “Root” folder we have 10 **RootMotion** Animation assets. Select All of the 10 **InPlace** Animation assets:



Picture 11: Adding Animation Modifier for all selected Animation assets

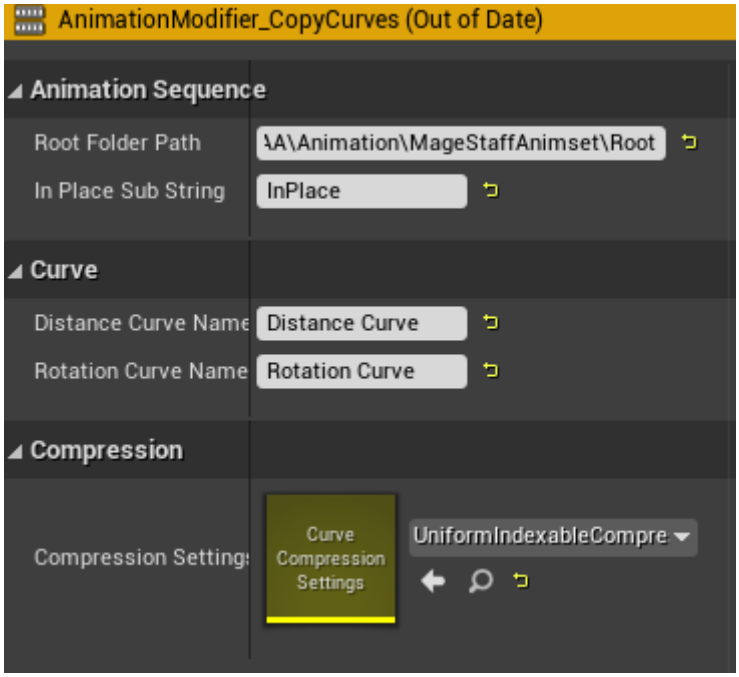
AnimationModifier_DistanceCurve will create Distance Curve for all of the selected **RootMotion** animations and we can use the same procedure but with AnimationModifier_CopyCurve and **InPlace** Animation assets to copy these created **RootMotion** curves to the **InPlace** Animations.

Let`s talk about the AnimationModifier_DistanceCurve for a while. As you can see, we have the Root Name as our first parameter. It is the Root Socket Name of the bone. It is “Root” by default. Next Part contains the selection of “Start”, “Stop” and “Rotation” substrings. What does that mean? Let`s take a look at the **Name** of the animations. “MageStaff_**Rotate**_Left_180”, “MageStaff_Run_Bwd_**Start**”, “MageStaff_Run_Left_**Stop**_RU”. Now, it is clear, isn`t it? When we choose all animations assets and apply the Animation Modifier to all of them at once, the system has to know, what kind of Animation Curve It has to create for every animation asset. The Substring in the name that the publisher uses to differentiate their animation assets will also be

the substring we are going to use to differentiate the animation assets while creating the curves. You can then choose any name you want for your curves. **But make sure that the curve name slots in the Distance Matching Component parameter will be same!**

After we have obtained all our **RootMotion** curves, now we have to find a way to copy all these curves to their **InPlace** versions. So, let`s continue!

2.AnimationModifier_CopyCurves

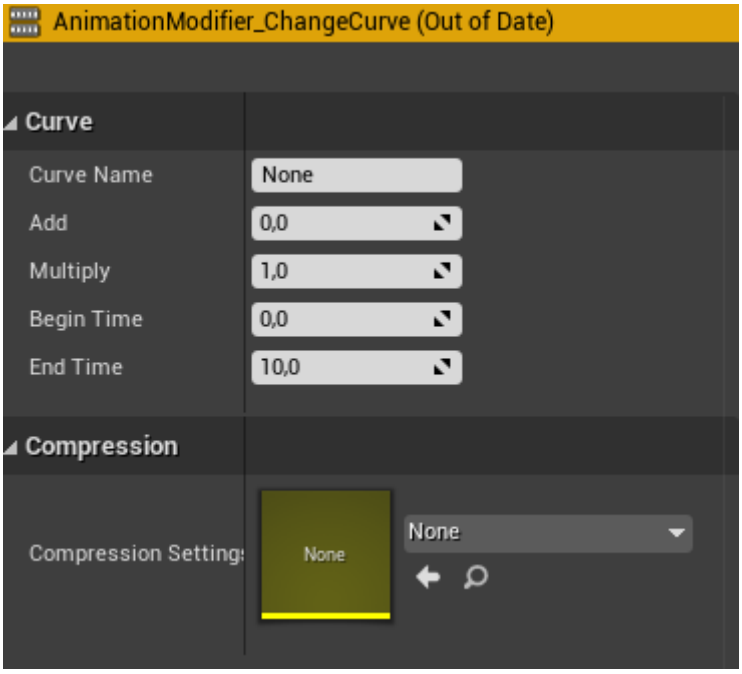


Picture 12: AnimationModifier_CopyCurves

Let`s take a look at one of the **RootMotion** animation asset`s name and it`s **InPlace** counterpart. “MageStaff_Run_Bwd_Start” and “MageStaff**InPlace**_Run_Bwd_Start”. Usually all animations have a substring that indicate they are **Inplace** versions. When we delete this substring we should get the exact same name. When we provide the Root Folder path, in which we have all our **RootMotion animation assets with curves**, we can then get the animation asset correctly in C++ and we know which animation to copy the curve, because they will have the exact same name. This lets us to create all the distance curves at once and after that to copy all the curves to the Inplace versions again at once. **While using AnimationModifier_CopyCurves don’t forget to compress the Animations with the UniformIndexable**!

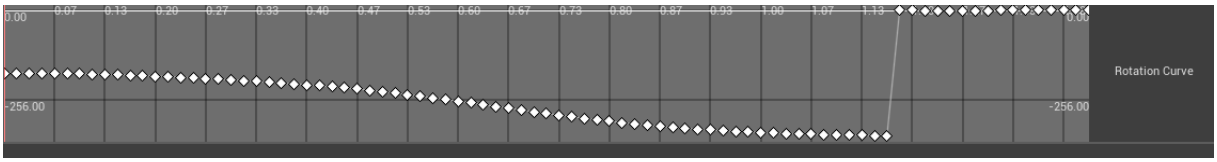
I want to introduce you to one last useful tool!

3.AnimationModifier_ChangeCurve

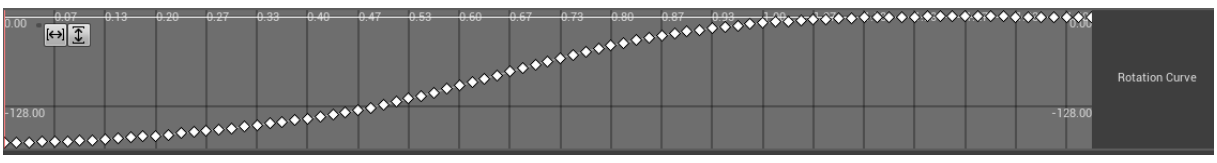


Picture 13: AnimationModifier_ChangeCurve

With this Animation Modifier you can change the curve values! All of them, or some of them. You can either multiply with or add a certain value. Here is a before and after example:



Picture 14: Before AnimationModifier_ChangeCurve Application



Picture 15: After AnimationModifier_ChangeCurve Application

Troubleshooting

- *Animation Does Not Play Correctly or Play At All!:* This issues is most of the time caused by providing incorrect curve information.
    - Check if your curve`s name is same with the name in the Actor Component.
    - Check if you have compressed the Animation that has the curve with the "UniformIndexable".
    - Check that your animation has the curve with the correct pattern.
    - Control if you get the correct times with checking the output of the respective "Curve Times" array of the Distance Matching Component.

- *There is a Stutter while Transitioning from Start to Looping State:* This is mostly caused by the lack or incorrect usage of **Sync Markers**!
    - Make sure that you have placed the Sync Markers correctly in all Start and looping jog animations (play with the location until you find a working spot) and that you have set up the Sync Groups appropriately in the AnimGraph.

- *Start Animations Play Way Too Fast:* In order to match the character`s speed, with high acceleration Start animations will play faster.
    - To avoid that you can either lower the Acceleration of your character through Character Movement Component or use a Speed Warping node to make it look more natural.

- *During the Stop Character Comes to A Complete Stop and The Animation Does Not Continue Playing:*
    - Change the value of the "Stop Tolerance" parameter in the Distance Matching Component.

- *While Applying Animation Modifier, I get errors:*
    - Don`t apply the same Animation Modifier to the same Animation Sequence twice. Delete the sequence and start from the start.

Summary

For more, please reach out to me through my given communication channels in this plugin`s description page. Thank you!