Unreal Engine 5.3 Release Notes

Overview of new and updated features in Unreal Engine 5.3

Unreal Engine 5.3 brings further improvement to the core UE5 toolset. This release delivers improvements in a wide variety of areas, including Rendering, Worldbuilding, Procedural Content Generation (PCG), Animation and Modeling tools, Virtual Production, Simulation, and more.

This release includes improvements submitted by our community of Unreal Engine developers on GitHub. Thank you to each of these contributors to Unreal Engine 5.3:

909185693, 34Pibo, aaronmjacobs, Acren, aijkoopmans, AishaBrown-SMG, aknarts, alexey-pelykh, alwintom, AndreaCatania, AndTikhomirov, ArcEcho, astutejoe, avudnez, baronpumpky, BenVlodgi, BinaerBoy, Bioliquid, BlenderSleuth, bretternst, brycehutchings, c4augustus, CaptainHoke, chalonverse, chrismcr, Cleroth, cneumann, crobertson-d3t, crssnky, David-Vodhanel, dbsigurd, Deathrey, DecoyRS, djethernet1, DomCurry, dorgonman, Drakynfly, DreikaEru, drichardson, dulanw, Dumbeldor, dyanikoglu, Edstub207, erebel55, error454, fieldsJacksonG, FineRedMist, flibitijibibo, foobit, freezernick, gbxAlx, gonfva, hellokenlee, hkzhugc, HSeo, Ilddor, ilkeraktug, ImaginaryBlend, iniside, jackatamazon, JakobLarsson-Embark, jammm, janurbanech13, jcb936, JDCruise, jfgoulet-gearbox, jimsimonz, Johnsel, JonasKjellstrom, jorgenpt, KacperKenjiLesniak, KaosSpectrum, KasumiL5x, KeithRare, kimixuchen, kissSimple, kniteli, KristofMorva, ldisthebest, LennardF1989, Levil0229, lflecunneen, lightbringer, lijenicol, lijie, liuhao0620, loening, LtRandolph, lucyainsworth, MalikuMane, mamoniem, manugulus, marcussvensson92, mariuz, mattiascibien, MaximeDup, microsoftman, MikBou123, mkotlikov, muit, OskarHolmkratz, pepsiandmilk, PhilBax, Phyronnaz, pirskij, praisesolek, projectgheist, QRare, qwe321, RandomInEqualities, Rareden, robert-wallis, RobertVater, roidanton, ryanjon2040, samhocevar, satoshi-iwaki, scorpio20t, Skylonxe, slackba, slonopotamus, Sn-a-ke, SRombauts, steelixb, stlnkm, teessider, tehpola, TheJamsh, TheoHay, Thomasleeee, tianyu2202, tommybear, tonetechnician, ToniSeifert, tuxerr, ukustra, Ultrahead, ungalyant, user37337, Vaei, velvet22, vorixo, Voulz, vsrc, WarmWar, WillmsBenjamin, WinsonSS, xiexbmu, yaakuro, yatagarasu25, yehaike, ZioYuri78, zompi2

Animating in Engine[1]

API Change:

- Fix inverted tangents on the Y axis for 2D controls.

[1] This is an incomplete document as I did not include the entire release. I just listed the sections I contributed to

New:

- Added a new `ControlRig.Sequencer.AutoGenerateTrack` console variable to toggle whether control rig tracks should automatically be created for new components.
- Control Rig FBX animations can now be exported and exported between Unreal

Improvement:

- Curve Editor filters have been improved and can now be set using ranges rather than individual selections, so you can now bake a range of frames or times.
- Curve proximities can now be updated when there is a drag operation so that the curve tooltips show the proper time and value.
- Curve Editor selections are now synchronized whenever new items are added to the editor.

Bug Fix:

- Fixed an issue that could occur by delaying a level sequence asset's creation until the tick proceeding the control rig actor is dropped into the viewport. This prevents the level sequence asset creation to be in the same transaction as the create actors transaction. Previously, the movie scene could be collected as garbage when undoing the action, because its creation took place during another transaction.
- Fixed an issue that could cause a memory leak with the backbuffer Render Target not being released at the end of a render.
- Constraints: propagate offset change so that local transform animations are synchronized.

Gameplay

New:

- You can now use the new experimental **Dead Blending** AnimGraph node. This node is designed as a drop-in replacement to the **Inertialization** node but uses the new robust "Dead Blending" algorithm which is more equipped to handle multiple simultaneous Initialization requests or when the Initialization transition is between two very different poses.
- Exposed a new method to blueprints that you can use to get the weight of a montage playing on a specified animation slot.

- Added the following AnimNode functions: `On State Exit`, `On State Entry`, `On State Fully Blended In`, and `On State Fully Blended Out`. These are located in any Animation State Graph's Result Node.
- Added support for AnimNodes to show any bound functions and not have them hardcoded.
- Added a way for users to specify the string for the default binding name of a function member reference in a blueprint by using the "DefaultBindingName" metadata tag.
- Added support for animation sync groups when using initialization for a state machine transition.
- Added support for explicitly specifying the transitions trigger time when using state machine automatic transition rules.
- Added a context menu option to Animation Blueprints to create an Animation Template from the blueprint that can be used to override animations using Child Blueprints.

Improvement:

- The `PlayMontageCallbackProxy` can now be derived from, in order to support plugins that want to provide customized versions.

Bug Fix:

- Fixed an issue that could cause a crash when using the Initialization AnimGraph node when re-parenting the Animation Blueprint.
- Fixed standard blending not working as expected with sync groups.
- Improved stuck detection in Distance Matching
- Disabled option to duplicate a state from the animation blueprint editor's tree view.

Movie Render Queue

New:

- Fixed issue where default settings would get added to a job that already exists when clicking on the **Render Movie** button from **Sequencer**.
- Exposed UMoviePipelineImageSequenceOutput_EXR so C++ plugins can reference it.
- When using the **Render all Cameras** feature, **Spawnable** cameras are now skipped if their default spawn track value is false.
- Added support for camera rotation to be preserved during a **Panoramic Render**.

- Updated the Console Variables setting to display the cvars contained within each cvar preset.

Improvement:

- Added a workaround for the **UI Renderer** setting having issues with some Layout Widgets not liking the 720p PIE preview window.

Optimization:

- Optimized the multi-sample accumulator for RGBA images on machines with many cores.

Bug Fix:

- Fixed issue where canceling a job reporting as a successful job for some callbacks.
- Fixed issue where UWorldSettings::Min/MaxUndilatedFrameTime were being modified and not restored after a render.
- Fixed UI warning about losing unsaved changes to **Queues** when there weren't actually any changes.
- Fixed issue where `{camera_name}` resolved into `main_camera` when not using the **Render All Cameras** feature is animating onto this.
- Fixed data type issue when exporting ProRes with OpenColorIO enabled.

Rigging

API Change:

- Creation of buffers is all done within the data interface at runtime.

New:

- Added output bone transfer list for anim node. You can now limit the bones being driven by a **Control Rig AnimNode** in the **Animation Blueprint Graph**.
- Added half edge connectivity data interface.
- Added unit node to check if a shape is available in the shape libraries.
- Added support for uint32 types.
- Added Int32 to **Name** node (`IntToName`)
- Added feature to update the **Reference** pose of a bone directly in the **Control Rig Editor**.
- Added **LocalChainDynamics** function which creates simple chain dynamics for the given chain.

- New skin weight editing tool enabled editing weights by painting or directly on selected vertices.
- IK Rig/Retarget assets can now be created directly from Skeletal Meshes in the right-click menu.
- Performance improvement to retrieve a template from its arguments.
- IK Retargeter toolbar redesigned for instant access to common retarget settings.
- Improved physics constraint editing in the Physics Asset Editor viewport.
- Extracted all instance data from Control Rig virtual machine and moved it to the context data struct, as a step for stateless virtual machines.
- FBIK solver now supports animated translation at runtime. Squashing and stretching limbs can use FBIK.
- RigVM context now has to be passed as a parameter to the virtual machine functions, rather than being located in the virtual machine itself (a step towards stateless virtual machine).

Improvement:

- Skin weights on skeletal meshes can now be edited through a new API in BP or Python.

Bug Fix:

- Fixed issue when adding curves to the output skeleton from `USkeletalMergingLibrary::MergeSkeletons`.
- Fixed **Control Rig Blueprint** compilation error related to missing function definition after a **Control Rig Blueprint** containing function references is duplicated or renamed.
- Fixed issue caused by **Control Rig Editor Details panel** when exiting PIE while debugging a Control Rig instance.
- Fixed an issue where sometimes the constraint viewport widget would be drawn relative to the wrong body.
- Fixed an issue where pressing the keyboard modifier for editing the constraint's child frame (Alt + Shift) also activated the keyboard modifier for the Lock camera to widget feature (Shift) which caused the translation widget to move erratically when editing the child frame.
- Fix for Control Rig Editor issue when adding comments and using undo / redo.
- Fixed Control Rig compiler issue occurring when a specific function parameter usage is found on a rig, due to invalid remapping of inline function parameters.
- Fixed an issue in the RigVM compiler when a rig function has more than 255 operands.

- Fix IK Retarget automatic chain mapping mode "exact all"; now it does not reset non-exact matches to none.
- FBIK solver will now dynamically reinitialize if the number of input bones changes dynamically at runtime.
- Cleaned up Control Rig Python files to remove mixed tab/space error in runtime.
- RigVMController: Added an array upgrade to reroute nodes with no source or target links where the value is an array, so it can load some deprecated rigs.
- FBIK solver now works correctly with only one effector (regression from 5.2).
- Fix for issue when adding a pin on Sequence Node in Control Rig graph.
- Fixed memory leak in FBIK solver (regression from 5.2).
- Spherical Pose Reader node in Control Rig fixed so it no longer flips when the activation region fills the whole sphere.
- RigVM plugin now has a custom serialization object version.
- Fix for Control Rig Gizmos not working properly; after changing the asset references to Soft Object Pointers.

Deprecated:

- Deprecated all Execution data interfaces. To set the execution domain of a custom compute kernel, use the Execution Domain field in the details panels of the kernel node. Valid domains should appear in the dropdown menu after connecting either a data interface to the kernel's primary group or a component binding node directly to the primary group pin.

Runtime

API Change:

- Extended the ImmediatePhysics_Chaos API so that you can now create Immediate Physics Constraints without copying values from Constraint Instances using a new CreateJoint function that accepts a JointSettings object instead of a FConstraintInstance.

New:

- Added a new dedicated track for Initializations to the Rewind Debugger.
- You can now click on Animation node tracks in the Rewind Debugger and the node will be highlighted in the corresponding Animation Blueprint to improve debugging workflows.
- Added missing tooltips for animation state machine schemas.
- Animation Blueprint runtime errors are now logged in the PIE log, if they were found during a game session.

- Anim blueprint logging can now use tokenized messages to hyperlink an error to the node it came from.
- UAnimInstance::IsSlotActive() is now thread safe.
- Player nodes in blendspace graphs are now added to GraphAssetPlayerInformation.
- Reinstated some anim blueprint library methods which now use the Anim Pose API.
- New colors to distinguish state aliases and conduits.
- Some LOD bone generation functions from the SkeletalMeshComponent have been extracted so they can be used from plugins.
    - Updated the ComputeRequiredBones function to use the extracted functions.
- Added a delegate to the SkeletalMeshComponent, so external systems can be notified when the RequiredBones get recalculated for a LOD.
- Moved the SkeletionRemapping.h and SkeletonRemappingRegistry from private to public, to allow including these files in plugins
- AnimBoneCompressionCodec now has a new DecompressPose function taking a FAnimBoneDecompressionData parameter, with the decompression data in the following separate array formats: translations, rotations, scales3D.
- Added the ability to DoubleClick AnimSegmentsPanel, in order to jump to the target asset. You can also shift + DoubleClick to open it in a new window.
- Added a new compiler note and visual warnings to state machine transitions if the transition rule has logic but the flag for automatic rule based condition is set

Improvement:

- Added a new helper function to find all external saveable packages.
- Added current time accessors and library functions to Anim Notify events to allow for easy access to relative times and ratios.
- Sync markers can now be managed in the same way as notifies.
- Neural network mask curves are forwarded to RigLogic from both the Animation Blueprint node and Control Rig implementations.

Crash Fix:

- Fixed an issue that could cause a crash when reloading a Control Rig Animation Blueprint.
- Fixed an issue that could cause a crash when undoing or redoing an Animation Sequence edit while an async task is in flight.
- Fixed an issue that could cause a reinstancing crash when using a custom Anim Instance proxy as a linked AnimGraph.

- Fixed an issue that could cause a crash when calling `UAnimInstance::GetAllCurveNames` from worker threads.
- Fixed an issue that could cause a crash when compiling an Animation Blueprint during automated tests.
- Fixed an issue that could cause a crash when right-clicking a modified curve node in a template Animation Blueprint.
- Fixed an issue that could cause a crash when unloading animation assets in a game feature plugin that are referenced by editor thumbnails.
- Fixed an issue that could cause a speculative crash during automated tests.

Bug Fix:

- Fixed an issue that could cause a regression with curve inertial blending due to the curve refactor.
- Fixed an issue that could cause the start position to be ignored when using sync groups.
- Fixed an issue that could cause an ASan issue when unregistering property type customizations on shutdown.
- Fixed an issue that could prevent the creation of a child Animation Blueprint in the Content Browser.
- Fixed an issue that could cause a crash when opening a template Animation Blueprint with the Curves tab open.
- Fixed an issue that could cause meshes to be distorted when the Leader Pose nod.
- Fixed an issue that could cause blending to incorrectly skip blending in and out when using inertial blending curve filtering.
- Fixed an issue that could cause Anim Node Functions to not be called in versioned builds when multiple are bound to the same Animation Blueprint node.
- Fixed an issue that could cause cached poses to be invalid on subsequent evaluations of an AnimGraph.
- Fixed an issue that could cause stack corruption caused by calling an Anim Node Function with an altered signature.
- Fixed an issue that could cause out-of-bounds access in Control Rig curve I/O.
- Fixed an issue that could cause distorted meshes when assigning assets using skeleton compatibility.
- Fixed an issue that could cause curves to be missed intermittently in cooked content.
- Fixed an issue that could cause bone LOD filtering to display incorrect results during automated tests.
- Fixed an issue that could cause the Layered Blend per bone node to not work correctly with per-bone curve filtering.

- Fixed an issue that could cause out-of-bounds access in PoseAssets that have additional poses with their respective curves.
- Fixed an issue that could cause fired notifies to not be reached when looping by clamping remaining time.
- Fixed an issue that could cause the virtual bone menus to not appear when adding additional virtual bones.
- Fixed an issue with the Control Rig Anim Node that could cause the Rigid Body Anim Node in a post process AnimBP to stop working.
- Fixed an issue that could cause OOM issue on consoles by reducing default alignment requirement of memory allocations in RigLogic.
- Fixed an issue that could cause performance problems when multiple faces (DNAs) are loaded during gameplay.
- Fixed issue when attempting to collect sync markers in a montage with an invalid slot track index.
- Fixed `bCanUseMarkerSync` to default to true even when the animation asset did not have any sync markers.
- Fixed `bCanUseMarkerSync` being true for followers when all valid markers were filtered out.
- Fixed issue when linking/unlinking anim layers from UGameplayAbility::EndAbility.
- Animation editor now correctly applies a preview mesh from a compatible skeleton.
- Fixed issue when filtering preview mesh on template anim BPs.
- Fixed Anim BP asset definition correctly displays the option to create child Anim BP.
- Fixed issue when adding curves to anim sequences and immediately getting the bone pose.
- Fixed issue in UBlendSpace::TickFollowerSamples due to hard coded bLooping flag.
- Fixed NaN crash in FAnimNode_LinkedInputPose::Evaluate_AnyThread.
- Ensure a clamped normalized time stays within the 0-1 range in the Blend Space Evaluator.
- Fixed "Start" animation notify event for state machine transitions not being fired when bSkipFirstUpdateTransitions flag is set to false.
- Fixed an issue that could cause a reinstancing crash in linked anim layers or graphs.
- Fixed an issue that could cause use-after free when duplicating animation attributes.
- Fixed an issue that could cause a crash when deleting a sync marker at the SyncMarkers Window, when finding an AnimMontage during the process.
- Fixed a but that would only look at player controllers, and not something like AI controllers, when considering viewpoints for auto significance calculation.
- Fixed an issue that could cause the GetLinkedAnimLayerInstanceByGroup tp not work for non-implemented layers with no instance assigned.

- Fixed an issue that could cause the Details panel to disappear when making curve metadata edits.
- Fixed an issue that could cause the preview instance from not being re-selectable after selecting an external anim instance to debug.
- Added some missing definitions for the UAnimInstance's function GetRelevantAssetPlayerInterfaceFromState.
- Fixed an issue that could cause a crash in FMirrorSyncScope.
- Fixed an issue that could cause blendspace sample entries from expanding when a sample animation is changed.
- Fixed an issue that could cause a crash on loading an anim BP with pose hander nodes that require skeleton updates to their animations.
- Added a missing ENGINE_API tag to SkeletetonRemappingRegistry, as it is required to use it from outside of the engine
- Fixed an issue that could cause a crash when using an incorrect bone index type usage in UAnimationSequencerDataModel::GeneratePoseData.
- Fixed an issue that could occur by ensuring when transforming bone indices during animation compression.
- Fixed an issue that could cause an issue with the SkeletalMeshComponent ComponentIsTouchingSelectionFrustum, as it was returning false positives due to missing check.
- Fixed an issue that could cause issues with the FBX Import due to a name restriction not allowing bone names to start with a number at RigVM hierarchies.
- Fixed an issue that could prevent the Animation Sequencer from triggering notifies repeatedly when using an animation mode track set to Use Anim Blueprint in a Level Sequence, by adding a parameter to the SkeletalMeshComponent SetAnimationMode function that allows not re-initializing the AnimationScriptInstance.
- Fixed an issue that could cause a crash when using the PoseableMeshComponent, when the component is being incrementally unregistered and the unregistering process takes longer than the time limit set in the component.
- Fixed an issue that could cause a crash when using one of the Animation editors when closing one viewport while additional viewports are open.
- Fixed an issue that could prevent the BlueprintEditor from being refreshed after adding an animation layer node into an animation graph.
- Fixed an issue that could cause an issue during an AnimComposite animation notify extraction when playing in reverse order.

Deprecated:

- Added a deprecation warning for `AnimNode_StateResult.h` located in `AnimGraphRuntime` module since the file got moved to `Engine/Animation`.
- Removed the deprecated mirroring functionality from SkeletalMesh.

Sequencer

New:

- Added the ability to create Takes for Subsequences.
- You can now add comments for shots and Sequencer sections.
- Added support for tags when bindings are copied and pasted in Sequencer.
- Added the option to select the material parameter collection asset from the track in Sequencer.
- Reassigned the `LevelSequenceEditorSettings.h` to public.
- Added command bindings for composition overlays to the Cinematic Viewport.
- Exposed the ability to access setters and getters for audio section looping, suppress subtitles, and attenuation functions.
- The Sequencer's Key editor now edits selected keys if they are selected. Otherwise, it edits at the current time as usual.
- Bound recording commands in the Take Recorder now play in world space so that they work in Editor and in PIE.
- The Clock Source menu is now always visible even in SubSequences, but is only editable when the Sequence Editor is focused on the root sequence.
- Several improvements have been made to the Sequencer's Import EDL:
    - Added the support of KEY CLIP NAME
    - Sequencer now parses the timecodes from the end of the line.
    - If the EDL contains a shot multiple times, subsequent shots will be created in Sequencer when importing, corresponding to the EDL shots.
- You can now change the Sequencer folder color of multiple folders by multi-selecting.
- Sequencer Track filters are now saved in preferences.
- Added a new transaction for toggling Show Marked Frames Globally.
- Added support for a start offset to camera animations.
- Upgrade audio track to use the new ECS evaluation.
- Moved level-sequencer-specific editor code to the LevelSequenceEditor plugin.
- Moved sequencer customizations to the editor view-model.
- Add the following improved edge interaction for sections:
    - You can now use draw easing handles as rounded triangles sized to match the grip size, instead of the previous bitmap.
    - Easing handles are bigger, with a bigger hit-testing area to make selection easier.
    - Exposed key size through the section interface in addition to grip size.

- Added support for blend types and multiple rows for material parameter tracks (widget and MPC).
- Exposed camera animation parameter to detail views.
- String properties and String tracks now use the new ECS evaluation.
- Add ability to lock marked frames on the time slider.
- Register/unregister the camera animation stand-in object from its own module.
- Added new user-defined bindings to Sequencer. This change adds the ability to define custom logic, using the directory blueprint, for object bindings. Instead of binding to the usual possessable or spawnable, this will optionally run arbitrary BP logic defined by the user to resolve the bound object. This makes it possible, for instance, to bind to the player pawn, or any other actor that might not exist at edit time. The UI is similar to that of the event track, and therefore reuses the director blueprint endpoint picker. There is however a new blueprint extension to compile these functions into the spawnables and possessables, and of course a spot to call these functions when appropriate.
- Added a new key that you can use to allow the player to be closed.
- You can now expand skeletal animation sections back to the left.
- Added new feature: Smart Auto Key.
- Smart Baking for Constraint Baking, Space Baking, Snapper, Control Rig collapsing and Baking Transforms.
- Swap Root Bone on Track.
- Animation Mode: Disable selection when clicking on an axis, avoids losing selection when manipulating.
- Curve Editor: Make normalized mode scrollable and panable. This makes sure that if you frame to the normalized mode now it doesn't use the max/min values but defaults.
- Control Rig: Shift now moves camera to stay with Control Rig when moving.
- Curve Editor: Keep Output bounds the same even if curves are destroyed so you can select/deselect and keep their curve bounds.
- Take Recorder now supports up to eight audio input sources for multi-track audio recording.
- Added time scrubbing to Sequencer and Curve Editor to match Viewport behavior.
- Added native accessors for common float/color light properties to improve performance when setting properties following sequencer evaluation.
- Curve Editor: Added option to show constraint/space bars.
- Linked Anim Sequences are now editor only.

Improvement:

- Take Recorder Timecode setting names now default to the same as Anim Sequence so that aligning works.

- Removed the Sequencer's Add (+) button text rollover visibility. Now the text is a tooltip so that there's no overlap with key editor buttons.
- Added the Sequencer's **Browse to Actor** property back to the actor's right-click context menu.
- Control Rig Controls filter is no longer enabled by default in Sequencer. You can optionally enable it, and your preference will be saved between sessions.
- Trim and Preserve now calculate the correct transform when the attached component isn't the root component.

Crash Fix:

- Fixed an issue that could cause a crash when sorting sections on load.

Bug Fix:

- Fixed an issue that could prevent the ability to toggle on or off tags inside of shots and subsequences.
- Fixed an issue that could cause new keys to be added when attempting to update an existing key.
- Fixed an issue that could cause the group manager to not refresh when performing an undo or redo command.
- Fixed an issue that could prevent the group manager selection to not function properly with folder paths.
- Fixed an issue with path track evaluations that could cause component velocity to not be set properly when an object is moved by the path track.
- Fixed an issue that could cause regenerating waveforms during playback to not function correctly by adding a slight delay to eliminate hitches.
- Fixed an issue that could cause duplicate entries in the property menu.
- Fixed an issue that could cause Show Marked Frames Globally to not be marked as a UPROPERTY so that it gets serialized.
- Take Recorder: Fixed range for camera cut section so that it works for subsequences when recording cameras.
- Remove unnecessary creation of camera cut track and section because they're already created when a camera is added.
- Fixed an issue that could cause the animation sequence to be marked dirty, due to the spawnable being modified, if the sequence is evaluating.
- Fixed mute/solo nodes when hierarchy changes by tracking old and new nodes paths.
- Fixed issue when the widget is replaced and no longer valid.
- Fixed issue where duplicating keys is not undoable and doesn't update the UI properly.
- Fixed an issue by keying the material parameter collection instance's current value if it exists, rather than always just the default value.

- Fixed an issue by compensating for transform origin when adding an object's transform to Sequencer.
- Fixed track row names when row indices have been remapped or a new row has been added.
- Fixed an issue that could occur when pasting a possessable, by only parenting it to the pasted target if the possessable's class is not an actor, such as a component. This fixes an issue where an actor could be pasted into another actor.
- Fixed an issue by ensuring the CineCamera actor's name is unique when creating as possessable. Normally this is done in the actor factory but here we spawn the actor directly.
- Check for valid animation on skeletal animation FBX export.
- Fixed issue where calling scripting functions (AddKeyInChannel, RemoveKeyFromChannel, SetDefaultInChannel, ReoveDefaultFromChannel, SetTimeInChannel, SetValueInChannel) is not undoable and doesn't update the UI properly.
- Fixed an issue by initializing layers for spawned actors. This fixes issues where if you add a spawnable to a layer, hide the layer, and then reopen the sequence in a new level, the spawnable won't be visible and won't respond to layers in the new level.
- Fixed an issue that could prevent Snap Sections in the Timeline by using the Source Timecode so that it takes tick resolution into account
- Fixed an issue by preventing all camera actors from being selected in the camera cuts menu even if it is hidden.
- Fixed an issue that could cause small sections to result in negative time-to-pixel conversion.
- Fixed an issue by carrying over the previous view target from a previously playing level sequence, if any, so that the last sequence played restores the actual original view target, and not the previous sequence's camera. This fixes issues with stuck cameras when some sequences are played on top of each other.
- Fixed an issue that could cause a crash when using the context menu in the Camera Cuts track to try and select an unspawned spawnable.
- Fixed an issue by preventing global custom accessors from being unregistered on shutdown.
- Fixed an issue that could cause recycling of view-models when track layouts are rebuilt. This fixes some UI issues with keys and sections being moved between track rows.
- Fixed an issue that could cause a crash when shakes are previewed in Sequencer or in the Camera Shake Previewer tool.
- Fixed an issue that prevented initial material parameter values to be cached correctly.
- Fixed an issue that could cause a crash when recompiling a user widget while Sequencer is open.

- Fixed an issue that could cause the Snap to Interval tool to not function when **Snap to Keys and Sections** was not selected.
- Fixed an issue by restoring aspect ratio axis constraint directly on the player if the level has changed during sequence playback.
- Fixed an issue by correctly restoring the Sequencer editor playback speed on stop and pause operations.
- Fixed an issue that could cause the key editor widget to not be cleaned up correctly when a track goes from 1 to 2 or more rows.
- Fixed an issue that could cause a crash when dragging an item from the Place Actor menu over a sub-scene section.
- Fixed a UI issue where the preview for the section length shown was only as long as the first sub-sequence asset, not the longest asset in your drag or drop selection.
- Fixed an issue that could cause Sequencer runners to detach and attach to linkers in interwoven ways.
- Fixed an issue that could cause the **Additive From Base** blend type to be missing an icon in the UE5 style.
- Fixed an issue by taking play-rate into account for blend times between blending camera cuts.
- Fixed an issue that could cause a Control Rig Component to be constantly reinitialized when used in Sequencer.
- Fixed an issue that could cause incorrect blending out of sequencer camera shake patterns.
- Fixed an issue by not calling the NotifyMovieSceneDataChanged() function when pasting tracks. This fixes an issue with Niagara not setting up its data correctly after pasting a track.
- Fixed an issue that could cause interactions to be incorrectly handled between the normal track area and the pinned track area.
- Fixed an NVVM refresh issues that could cause view-models who already have a recycle list would not recycle their children. This meant that a re-used track-row view-model would not get its children cleared out before it is rebuilt with new underlying data.
- Fixed an issue with the Channel Curve Models in Sequencer Curve Editor that could cause the tracking interpolation type of neighboring keys to not operate properly when a new key was added.
- Fixed an issue that could cause the MovieSceneSequenceExtensions::GetBindingId to not be properly exposed to python scripting.
- Fixed an issue that could cause ease-in curves to not show properly when a section starts at time 0 that also does not have an ease-out.
- Fixed an issue by not caching pre-animated state on entities that are being torn down.
- Fixed an issue that could cause BOOL and INT curves to not be drawn past the left or right-most key value in Curve Editor.

- Fixed an issue that could cause a null check in Movie Scene playback.
- Fixed an issue that could cause autosizing of subsections, audio sections, and skeletal animation sections, to not function properly, as they were all missing taking into account start or end offsets.
- Fixed an issue that could cause a spawnable binding to be overridden during the sequencer, the spawned actor will get left behind, creating duplicate characters. Now, when a binding is invalidated, and we detect an overridden binding, Sequencer checks if a spawnable is registered, and if so, destroys that spawnable.
- Fixed an issue by setting a zero value on an additive Control Rig section, so it doesn't always add and uses default interpolation.
- Fixed an issue by adding the sequencer ability to change the SkeletalMesh asset from a Skeletal Mesh Component in cooked builds.
- Snapper: Use default key type when setting keys.
- When changing time via translate hotkeys get all changes first, then apply them to avoid double transforms with space/constraint keys.
- Control Rig: When looking for a binding that constrains a Control Rig, it takes into account the Control Rig so it picks the correct Actor or its component. For certain setups the face and body rig both bind to the same component on the Control Rig, but bind differently on the Sequencer side.
- Fix for missing or incorrect timecode values being stored in audio assets when recording with Take Recorder.
- Constraints baking: changed the tangent type to the default for the keys at compensation to avoid inserting keys that may have flat tangents.
- Control Rig: Fix keying weighted additive sections.
- Fix copying actors in Sequencer that live in sublevels, you can end up selecting the wrong Actor.
- Only sync curve selection when updating the tree if we have no curves, otherwise this will blow away what the animator is working on.
- Fixed silent failure case in Sequencer when scrubbing audio at low frame rates.
- Fixed an issue that occurred when applying an FFT filter with insufficient number of keyframes selected in Curve Editor.
- Calling PasteBindings() from Blueprints or Python correctly returns the newly created bindings.
- Linked Anim Sequences work with nested sequences.
- Curve Editor: Don't show the space or constraint key bar spaces unless there is a space.
- Snapper: Do not include controls without shapes when snapping to avoid issues.
- Animation Mode: Restore Gizmo Scale when leaving/entering Animation Mode.
- Linked Anim Sequence: Only dirty if the data is changed, not if refreshing or changing sequences.

- Control Rig: Make sure it is on game thread when handling initialize events, fix possible issues with procedural rigs.
- Fixed Sequencer player playback replication synchronization to only happen once per tick.

Deprecated:

- Deprecated the GetBoundObjects as it is incompatible with Spawnables. It is recommended to use ULevelSequenceEditorBlueprintLibrary::GetBoundObjects because it operates on the currently opened sequence in Editor and can return valid spawnable objects.
- Remove Matinee's PostInterpChange, as it is not often used or compatible with projects.

Synthesis

New:

- Added progress bars when creating and compressing Morph Targets.
- Added support for using Physics Blend Weight alongside Physics Control Component.
- Added a new `UpdateKinematicFromSimulation` flag to BodyInstances. This allows kinematics to be treated the same as dynamic objects when simulating with async physics. The flag defaults to `false`, so there is no change in the default behavior, unless using the Physics Control Component, which sets the flag by default.
- You can now get the current or target transforms with Physics Control Component.
- You can now set the Physics Control Component to use manual updates.
- The Physics Control Component damping ratio in the PhysicsControlMultiplier struct has now been exposed.
- You can now bulk edit Physics Control Component settings, such as the target positions and orientations.
- You can now reset the physics transforms to the cached targets when using a Physics Control Component.
- Added the ability to use profiling mark up when using a Physics Control Component.
- Removed the NNI from ML Deformer VertexDeltaModel and NearestNeighborModel.
- Fixed an issue that could cause a crash when initializing the ML Deformer on Linux operating systems.

Improvement:

- Improved ML Deformer reconstruction quality:
    - The ML Deformer reconstruction method now uses a smoothed L1 loss function, adding a new beta **Smoothing Loss** property to the training settings.
    - The reconstruction method also now uses cosine annealing learning rate scheduler, which removes the Learning Rate Decay setting.
- The ML Deformer now displays a warning when using the Vertex Delta Model, that it should not be used in production.
- The ML Deformer now uses extra **Get functions** to get private members.
- Verbose logging has been added where non-verbose logging wasn't needed.
- Added additional null safety checks to the ML Deformer.
- The ML Deformer now prevents sampler reinit when not needed.
- ML Deformer models can now specify the number of floats per bone.
- The ML Deformer now detects changes in vertex count, and shows a warning in the morph target section, when dynamic Morph Target updates aren't possible until after retraining.
- Improvement the cleanup of the external Morph Targets, which also clears out the weight data.
- Improved the component reinits by reducing the number of units.
- Improved the rendering of bones inside the ML Deformer asset editor.
- The Initialise Physics Control Component control strengths are now set to `0` by default, instead of `1`.

Crash Fix:

- Fixed an issue that could cause a crash by preventing the ML Deformer from updating dynamic Morph Targets when it is not possible
- Fixed an issue that could cause a crash when the local bone transforms on the skeletal mesh component were empty.
- Fixed an issue that could cause a crash when creating a new actor blueprint, adding the blueprint to a skeletal mesh and ml deformer component, then setting a neural morph model as ML Deformer asset, and selecting the vertex delta model mesh deformer in the skeletal mesh component.

Bug Fix:

- Fixed an issue that could cause the ML Deformer asset to behave incorrectly. This could likely happen when aborting during training, or changing assets after training, and saving. Now, the editor correctly sets section indices inside the Morph Targets, rather than just including all sections and sorts the Morph Target deltas by vertex number.

- Fixed an issue that could cause unexpected behavior when using curves in the Local Model.
- Fixed an issue that could cause the ML Deformer editor to enter a state where the timeline does not function properly after changing the input of an asset, such as a property on a skeletal mesh.
- Fixed an issue that could cause compatibility checks to fail after training.
- Fixed an issue that could cause a check to fail in debug mode.
- Fixed an issue by setting the Physics Blend Weight property to 1 by default when using a Physics Control Component to simulate physics. This ensures that using a mixture of kinematic and dynamic bodies works with skeletal meshes.
- Fixed an issue that could occur where the ML Deformer time sampling was incorrect.
- Fixed an issue that could occur when render graph errors in ML Deformer Vertex Delta Model.
- Fixed some issues that could occur with ML Deformer Vertex Delta Model inferences.
- Fixed an issue that could occur with the VertexDeltaModel which caused animation to appear frozen.
- Fixed an issue that could cause a module in ML Deformer to be missing.
- Fixed an issue that could cause an incorrect mismatched vertex error in MLDeformer.

Removed:

- Removed a substring redirect from the ML Deformer, which wasn't needed anymore and triggered a warning in the log.