COMPLETE USER GUIDE , VERSION 3.0

Scar Rewind System

Unreal Engine 5.2 , 5.7 , Zero Dependencies , Crash-Free

Per-Actor Rewind	**Global World Rewind**
Ghost Trail (Blueprint Actor)	**Ghost Trail Linger Duration**
Pickup-Safe Global Rewind	**Rewind Energy System**
Custom Variable Tracking	**Named Bookmarks**
Post Process Hook	**Axis Lock , Cooldown**

Blueprint-first design , Zero C++ required , Drop-in to any project , Fab Marketplace Ready ,

SECTION

Table of Contents

SECTION 1

What Was Fixed in Version 3.0

Version 2.0 compiled without errors but crashed Unreal Engine 5.2 at runtime. Eight root causes were identified and resolved. Every fix is documented below for full transparency.

FIX 1 , Raw AActor* map key, Crash at Actor Destruction

The global rewind cache used TMap. When any tracked Actor was destroyed mid-session the raw pointer became dangling, causing an access violation on any subsequent map access. Fixed by switching to TMap with TWeakObjectPtr inside the record. FObjectKey is GC-safe and never dangling.

FIX 2 , FWorldDelegates::OnWorldPreActorTick does not exist in UE 5.2

This delegate was added in a later engine version. Using it in UE 5.2 produced a link error at best and undefined behaviour at worst. Fixed by inheriting the subsystem from FTickableGameObject and implementing Tick(float), which is stable across all UE 5.x versions.

FIX 3 , TActorIterator run during world transitions

The global recording timer fired on a fixed interval regardless of world state. If it fired during level streaming or world teardown, iterating actors in a partially destroyed world caused crashes. Fixed by accumulating delta time in FTickableGameObject::Tick and guarding with WorldType checks.

FIX 4 , NewObject without proper attachment

The post process component was created with NewObject and RegisterComponent but never attached to the owner's scene hierarchy. UE's cleanup during PIE stop could not find a valid outer and assert-crashed. Fixed by calling NewObject with the owner as outer, then AttachToComponent and RegisterComponent in the correct order with a unique generated name.

FIX 5 , Ghost Trail spawning AActor base class directly

The old ghost trail tried to spawn a bare AActor or AStaticMeshActor and configure it from C++, which was brittle and inflexible. Replaced entirely by a TSubclassOf Blueprint class picker. Developers assign any Blueprint Actor as the ghost. Full creative control, zero engine-internal assumptions.

FIX 6 , Deinitialize() calling GetWorld() on a null world

During editor shutdown and level streaming, GetWorld() can return nullptr in Deinitialize(). The old code accessed the timer manager through that null pointer. Fixed by removing all GetWorld() calls from Deinitialize(). Cleanup now only resets flags and empties arrays, which is safe without a valid world.

FIX 7 , GlobalActorCache iteration with invalidated keys

Related to Fix 1. Raw pointer keys could become stale between recording ticks. The fix adds an explicit stale-entry purge pass using TWeakObjectPtr validity checks after each recording cycle.

FIX 8 , StopGlobalRewind called from inside the playback iteration loop

When all actors reached the start of their buffers, StopGlobalRewind() was called directly from inside the for-loop iterating GlobalActorCache. This modified the map while it was being iterated, corrupting the hash table. Fixed with a bPendingGlobalStop deferred flag that Tick() processes after the loop completes.

SECTION 2

Quick Start , Blueprint Setup

The Scar Rewind System is fully Blueprint-native. Every feature is accessible through Blueprint nodes. No C++ is required at any stage.

1

Enable the Plugin

Go to Edit, then Plugins. Search for Scar Rewind System. Check Enabled. Click Restart Now. The plugin appears under the Gameplay category.

2

Add the Component to Your Actor

Open any Blueprint Actor. In the Components panel click Add Component and search Scar Rewind Component. Select it. Recording begins automatically when you press Play, no extra setup needed.

3

Wire Up Input, Start Rewind

In the Event Graph, right-click and add your Input Action event (for example, IA_Rewind). Drag the Scar Rewind Component reference into the graph. Connect the Started or Pressed exec pin to the Start Rewind node.

4

Wire Up Input, Stop Rewind

From the same input event, connect the Completed or Released exec pin to the Stop Rewind node on the component. That is the complete minimal setup.

5

Test In-Editor

Press Play. Enable bShowDebug on the component to see a live status overlay floating above the Actor. It shows state, buffer fill, and energy. Hold your rewind key to see the Actor travel back in time.

TIP Enhanced Input is recommended but not required. The component works with any input method, including legacy Input Action, keyboard events, or custom input systems.

SECTION 3

Creating Your Ghost Blueprint Actor

The ghost trail spawns a Blueprint Actor you create and fully control. This gives you complete creative freedom. Any mesh, material, particle effect, or animation can serve as the ghost echo. The component handles spawn, position updates, and destruction automatically.

Step-by-Step Ghost Setup

1 **Create a New Blueprint Actor**
In the Content Browser right-click, choose Blueprint Class, then Actor. Name it something like BP_RewindGhost.

2 **Add a Static Mesh Component**
Inside BP_RewindGhost, click Add, then Static Mesh Component. Assign the same mesh your character or prop uses.

3 **Apply a Translucent Material**
Create a Material with Blend Mode set to Translucent and Lighting Model set to Unlit. Set Opacity to around 0.35. Apply it to the mesh. You can also add a Niagara component for a particle-based echo instead of or alongside the mesh.

4 **Disable Collision**
Select the mesh component. In Details, Collision, set Collision Presets to NoCollision. This prevents the ghost from blocking the player or other actors.

5 **Optional, Add Effects**
Add Niagara components, Timeline animations, audio, material animations, or any Blueprint logic you want. The ghost Blueprint can do anything a normal Blueprint Actor can do.

6 **Assign in the Rewind Component**
Select your Actor that has the Scar Rewind Component. In Details, Scar Rewind System, Ghost Trail, set Ghost Actor Class to BP_RewindGhost and enable bShowGhostTrail.

7 **Set the Linger Duration**
In Details, Scar Rewind System, Ghost Trail, set Ghost Trail Linger Duration to how many seconds the ghost should stay visible after rewind ends. The default is 1.5 seconds. Set it to 0 to destroy the ghost immediately when rewind stops.

NOTE The component automatically spawns one ghost instance when rewind starts, moves it to match the rewound position every frame, and starts the linger timer when rewind ends. You do not need any Begin Play or construction logic in the ghost Blueprint for the basic trail to work.

TIP For a dissolve effect, add a Timeline in the ghost Blueprint that fades opacity from 0.5 to 0 over the same duration as your Ghost Trail Linger Duration. This makes the ghost fade out gracefully instead of disappearing abruptly.

SECTION 4

Component Properties Reference

All properties are editable from the Details panel in the Blueprint editor. No code is required to configure any of these settings.

Core

Property	Default	Description
Buffer Duration	10 s	Seconds of history stored. Raise for longer rewinds, up to 60 seconds.
Recording Interval	0.05 s	Snapshot frequency. 0.05 seconds equals 20 snapshots per second.
Rewind Speed	1.5x	Playback speed multiplier. 1.0 equals real time, 2.0 equals twice as fast.
Location Threshold	2 cm	Minimum movement before saving a snapshot. Skips idle frames.
Rotation Threshold	0.5 deg	Minimum rotation before saving a snapshot.

Energy System

Property	Default	Description
Use Rewind Energy	Off	Enable built-in stamina. Rewind stops automatically at zero energy.
Max Rewind Energy	100	Maximum size of the energy pool.
Energy Drain Rate	20 / s	Energy consumed per second while rewinding.
Energy Recharge Rate	10 / s	Energy restored per second when idle.
Min Energy To Start Rewind	10	Minimum energy required to begin a new rewind.

Cooldown

Property	Default	Description
Rewind Cooldown	0 s	Seconds before another rewind can start after one ends. 0 means no cooldown.

Axis Lock

Property	Default	Description
Lock Location X	Off	Prevent rewinding the X world-space position.
Lock Location Y	Off	Prevent rewinding the Y world-space position. Use this for 2D sidescrollers.
Lock Location Z	Off	Prevent rewinding height. Useful for top-down games.
Lock Rotation	Off	Prevent rewinding rotation. Position still rewinds normally.

Ghost Trail

Property	Default	Description
Show Ghost Trail	Off	Spawn the ghost Blueprint Actor during rewind playback.
Ghost Actor Class	None	The Blueprint Actor class to spawn as the ghost. Create your own translucent Blueprint and assign it here.
Ghost Trail Linger Duration	1.5 s	How long the ghost stays visible after rewind ends. 0 destroys it immediately. Positive values keep the ghost alive so players can see where they came from.

Post Process

Property	Default	Description
Use Post Process Effect	Off	Automatically activate a post process effect during rewind.
Post Process Chromatic Aberration	2.0	RGB fringe strength. Gives a time-glitch look. 0 means none.
Post Process Vignette	0.6	Screen-edge darkening during rewind. 0 means none.
Post Process Desaturation	0.8	Colour drain amount. 1.0 equals full grayscale.

Priority and Debug

Property	Default	Description
Rewind Priority	5	1 to 10. Higher priority means a larger buffer allocation in the global memory budget.
Show Debug	Off	Draws live state text above the Actor. Compiled out automatically in Shipping builds.

Property	Default	Description
Show Debug Path	Off	Draws the recorded movement path as coloured dots. Also compiled out in Shipping.

SECTION 5

Blueprint Node Reference

Every function listed here is available as a Blueprint node. Drag the Scar Rewind Component reference into the Event Graph and pull off a wire to search for any of these nodes.

Core Nodes

Function , Node	Category	Description
Start Rewind	Core	Begin rewinding. Blocked automatically if the buffer is empty, rewind is already active, cooldown is running, or energy is too low.
Stop Rewind	Core	Stop rewinding. Actor stays at the rewound position. Recording resumes automatically.
Clear Buffer	Core	Erase all history. Cannot rewind past this call. All bookmarks are also cleared.
Set Recording Enabled (bool)	Core	Pause or resume snapshot recording without affecting active playback.

Bookmark Nodes

Bookmarks save a named moment in history. Restore Bookmark is an instant teleport, it does not play through the intervening history, it jumps directly to that moment.

Function , Node	Category	Description
Save Bookmark (Name)	Bookmarks	Save a named moment at the current buffer head. Up to 16 per component. Re-saving the same name updates it.
Restore Bookmark (Name), returns bool	Bookmarks	Instantly move the Actor to the saved position and restore tracked variables. Returns false if the name was not found.
Delete Bookmark (Name)	Bookmarks	Remove a single bookmark by name.
Clear All Bookmarks	Bookmarks	Remove all bookmarks. Does not clear the snapshot buffer.

TIP Call Save Bookmark when the player enters a safe zone. On death, call Restore Bookmark to teleport back instantly with no animation required.

Energy Nodes

Function , Node	Category	Description
Get Current Energy, returns float	Energy	Raw energy value from 0 to Max Rewind Energy.
Get Energy Ratio, returns float	Energy	Energy as a 0 to 1 fraction. Wire directly into a Progress Bar Percent binding.
Set Energy (float)	Energy	Force-set the energy value. Clamped to 0 through Max Rewind Energy. Fires On Energy Changed.

TIP Drag the On Energy Changed event onto the graph and wire it into a Progress Bar Set Percent node. This updates your energy UI automatically without polling every tick.

State Query Nodes

Function , Node	Category	Description
Is Rewinding, returns bool	Query	True while rewind playback is active.
Is Recording, returns bool	Query	True while new snapshots are being saved.
Is On Cooldown, returns bool	Query	True while the cooldown timer is running.
Get Rewind State, returns Enum	Query	Returns Idle, Recording, Rewinding, or Paused.
Get Buffer Size, returns int	Query	Number of snapshots currently in the buffer.
Get Buffer Capacity, returns int	Query	Maximum snapshot capacity of the buffer.
Get Buffer Fill Ratio, returns float	Query	Buffer fill as 0 to 1. Use for a rewind-meter progress bar.
Get Recorded Duration, returns float	Query	Approximate seconds of history currently stored.
Get Cooldown Remaining, returns float	Query	Seconds until cooldown expires. Returns 0 when ready.
Get Last Snapshot Custom Floats, returns float[]	Query	Returns an array of the most recently recorded custom float values. Index matches registration order.

SECTION 6

Events and Delegates

All events are BlueprintAssignable. In the Blueprint editor, select the Scar Rewind Component in the Components panel, then look in the Details panel under Events and click the green plus button next to any event to add it directly to the graph.

On Rewind State Changed (bIsRewinding)

Fires when rewind starts (true) or stops (false). Use this to play sounds, trigger VFX, update UI, or lock player movement. This is the primary event for reacting to rewind in Blueprint.

On Rewind Buffer Exhausted

Fires when the buffer is fully consumed and rewind auto-stops. Show a no-more-rewind notification here, or trigger a screen shake or sound.

On Energy Changed (CurrentEnergy)

Fires whenever energy changes by more than 0.01. Bind a Progress Bar to this instead of polling Get Energy Ratio every tick. This event fires during both drain and recharge.

On Bookmark Saved (BookmarkName)

Fires when Save Bookmark succeeds. Use this to show a checkpoint-saved notification or play a confirmation sound.

On Global Rewind State Changed (bIsActive) , Subsystem

Fires on the Scar Rewind Subsystem when global world rewind starts or stops. Get the subsystem using Get World Subsystem, then click plus next to this event in Details. Use it to apply Set Global Time Dilation or play a world-wide rewind sound.

NOTE To bind an event in Blueprint: select the component in the Components panel, open the Details panel, scroll to the Events section, and click the green plus button next to the event you want. Unreal creates the event node in the Event Graph automatically.

SECTION 7

Global World Rewind

Global World Rewind rewinds every eligible movable Actor in the level simultaneously with a single Blueprint node. Enemies, projectiles, props, and vehicles are all rewound together. Actors with a Scar Rewind Component use their own per-actor buffer. All other eligible movable Actors use the global snapshot cache built by the subsystem.

How to Access the Subsystem in Blueprint

Right-click in any Blueprint Event Graph and search for Get World Subsystem. In the Class dropdown select Scar Rewind Subsystem. Pull off the return value wire to access all global rewind nodes.

TIP Store the subsystem reference in a local variable to avoid repeating the Get World Subsystem call across multiple nodes.

Global Rewind Nodes

Function , Node	Category	Description
Start Global Rewind (Duration)	Global	Rewind every eligible movable Actor. Duration 0 uses the full buffer. Duration 3.0 rewinds exactly 3 seconds back.
Stop Global Rewind	Global	Stop global rewind. All Actors resume from their rewound positions.
Is Global Rewinding, bool	Global	True while global rewind is active.

Global Rewind Settings (on the Subsystem)

Property	Default	Description
Global Buffer Duration	10 s	History kept for Actors without a Scar Rewind Component.
Global Record Interval	0.05 s	Snapshot frequency for the global cache.
Global Rewind Speed	1.5x	Playback speed for all Actors during global rewind.

Excluding Actors with the ScarIgnoreGlobal Tag

To exclude any specific Actor from global rewind, add the tag ScarIgnoreGlobal in its Details panel under Tags. Actors with Mobility set to Static are always excluded automatically because they cannot move. See Section 8 for full pickup exclusion details.

NOTE In large scenes with many dynamic Actors, raise Global Record Interval to 0.1 to reduce CPU cost. Tag non-essential background props with ScarIgnoreGlobal.

SECTION 8

Pickup Exclusion from Global Rewind

Global rewind does not affect pickups. Rewinding the world while the player has already collected an item would cause confusing behaviour, for example a weapon disappearing from the player's hand or a health pack reappearing and then vanishing again as the timeline moves. The system provides two automatic methods to keep pickups out of the global rewind.

Method 1 , Tag the Pickup Actor (Recommended)

Add the tag ScarIsPickup to any Actor's Tags array in the Details panel. The global rewind system will skip this Actor entirely, regardless of whether it has been picked up or not. Use this for weapons, health packs, ammo, keys, collectibles, and any other item the player can interact with.

Select the pickup Actor in the level or open its Blueprint.

In Details, scroll to Tags.

Click the plus button and type: ScarIsPickup

That is all. The Actor is now permanently excluded from global rewind.

Method 2 , Ownership Detection (Automatic)

When a player picks up an item, most inventory and equipment systems set the item's Owner to the Pawn that collected it. The global rewind system detects this automatically. Any Actor whose Owner is a Pawn subclass is skipped during global rewind. This means already-collected items are excluded without any extra tags or configuration.

NOTE If your pickup system does not set the item's Owner, use Method 1 (the ScarIsPickup tag) as the reliable fallback. Both methods can be used together on the same Actor.

Summary of All Exclusion Rules

Condition	Result
Actor has Mobility = Static	Always excluded. Static Actors cannot move.
Actor has tag ScarIgnoreGlobal	Excluded. General-purpose opt-out for any Actor.
Actor has tag ScarIsPickup	Excluded. Designed specifically for collectible items.
Actor's Owner is a Pawn subclass	Excluded. Item has been picked up by a player or AI.
Actor has a Scar Rewind Component	Excluded from global cache. Uses its own per-actor buffer instead.
Actor is a pure logic Actor (AInfo subclass)	Excluded. These have no world presence to rewind.

IMPORTANT The ScarIsPickup tag excludes the Actor permanently, even before the player picks it up. If you want un-picked items to rewind with the world but picked items to stay in the player's hands, rely on Method 2 (ownership detection) only and do not add the tag.

SECTION 9

Batch Operations via Subsystem

The Scar Rewind Subsystem lets you control every registered Scar Rewind Component in the level at once. All components register automatically at Begin Play, you do not need to manage this manually.

Access via Get World Subsystem, Scar Rewind Subsystem

Function , Node	Category	Description
Rewind All Actors	Batch	Call Start Rewind on every registered component.
Stop All Rewinds	Batch	Call Stop Rewind on every registered component.
Clear All Buffers	Batch	Call Clear Buffer on every registered component.
Get Registered Component Count, int	Batch	How many Scar Rewind Components are currently active in the world.
Start Global Rewind (Duration)	Global	Rewind every eligible movable Actor. Duration 0 uses the full buffer.
Stop Global Rewind	Global	Stop global rewind. All Actors resume from their rewound positions.
Is Global Rewinding, bool	Global	True while global rewind is active.

SECTION 10

Game Design Recipes

Ready-made configurations for common rewind gameplay patterns. All settings are applied in the Details panel of the Scar Rewind Component or via Blueprint nodes. No code is required.

Death Rewind

Player dies. Call Start Global Rewind with Duration 3.0 to rewind 3 seconds. The ghost trail with a 2-second linger shows the player where they were. Post process desaturation signals the timeline reversal. The energy system prevents abuse.

Blueprint Config:

Use Rewind Energy, On

Show Ghost Trail, On, assign your ghost Blueprint, set Ghost Trail Linger Duration to 2.0

Use Post Process Effect, On, Desaturation 0.8

Call Start Global Rewind (3.0) from your death event

Puzzle Environment Reset

Every puzzle prop has a Scar Rewind Component. Player makes a mistake. Call Rewind All Actors on the subsystem to reset everything. Call Clear All Buffers after a successful solve to prevent rewinding past the clean state.

Blueprint Config:

Add Scar Rewind Component to every puzzle Actor

Call Rewind All Actors on mistake

Call Clear All Buffers on success

Bullet Time Sequence

Call Set Global Time Dilation (0.2) to slow the world, then call Start Global Rewind (2.0) to replay the last 2 seconds in slow motion. Chromatic aberration at 3.0 adds visual impact.

Blueprint Config:

Post Process Chromatic Aberration, 3.0

Post Process Desaturation, 1.0

Call Set Global Time Dilation (0.2) then Start Global Rewind (2.0)

Checkpoint Restore

Call Save Bookmark with name Checkpoint when the player enters a safe zone. On death, call Restore Bookmark with the same name to teleport back instantly.

Blueprint Config:

Call Save Bookmark (Checkpoint) at each safe zone trigger

Call Restore Bookmark (Checkpoint) from your death or respawn event

Restore Bookmark is instant. It does not play through the intervening history

Resource-Limited Rewind

A stamina-gated rewind. The player gets 4 seconds per use, takes 20 seconds to fully recharge, and must wait 2 seconds between uses. Bind the energy UI bar to On Energy Changed.

Blueprint Config:

Use Rewind Energy, On

Max Rewind Energy, 100 | Energy Drain Rate, 25 | Energy Recharge Rate, 5

Rewind Cooldown, 2.0

Bind On Energy Changed event to Progress Bar Set Percent

2D Platformer , Side-Scroller Lock

Prevents the rewind from pulling the character off the 2D gameplay plane or flipping the sprite rotation.

Blueprint Config:

Lock Location Y, On (locks depth axis)

Lock Rotation, On (keeps sprite facing correctly)

Collectible-Safe World Rewind

Player collects items during gameplay. A global rewind rewinds enemies and projectiles without touching any picked-up items. Pickups that have not been collected yet also stay in place if tagged.

Blueprint Config:

Add tag ScarIsPickup to every collectible Actor

Call Start Global Rewind normally

Picked-up items (Owner is Pawn) and tagged items are excluded automatically

SECTION 11

Performance Notes

The Scar Rewind System is built with performance as a first priority. Here is what you get out of the box and what to watch for in large scenes.

Tick Disabled When Idle

The component enables Tick only during active rewind, energy management, or debug display. A stationary Actor with the component attached and no active features has zero per-frame cost.

Timer-Based Recording

Recording uses a looping timer, not a per-frame Tick. Snapshots are taken at your configured Recording Interval regardless of frame rate. Frames between intervals cost nothing.

Zero Runtime Allocations

The circular buffer is pre-allocated once in Begin Play. Pushing a snapshot overwrites existing memory in place. No garbage collection pressure during play.

Movement Threshold Skips Idle Actors

Actors that have not moved or rotated beyond the threshold values do not write a snapshot on that recording cycle. Stationary Actors consume no buffer space.

FTickableGameObject Subsystem

The subsystem uses FTickableGameObject, not UActorComponent Tick. This avoids being blocked by ActorComponent tick prerequisites and runs more predictably across versions.

Ghost Linger Timer is Zero-Cost When Idle

The ghost linger timer is a single FTimerHandle. It is only active for the duration after rewind ends and costs nothing when not running. If Ghost Trail Linger Duration is 0, no timer is created at all.

Global Rewind at Scale

The global recording iterator runs at Global Record Interval, not every frame. In scenes with 200 or more dynamic Actors, raise the interval to 0.1 and tag non-essential background props with ScarIgnoreGlobal to keep CPU cost low. Pickup Actors tagged ScarIsPickup are skipped entirely and add no recording overhead.

SECTION 12

File Reference

All plugin source files are in your project's Plugins/ScarRewindSystem/ folder. You do not need to edit any of these files to use the plugin. This reference is for developers who want to extend the system.

File	Purpose
ScarRewindSystem.uplugin	Plugin descriptor, version info, and Fab marketplace metadata.
ScarRewindSystem.Build.cs	Module build rules. Includes PhysicsCore and RenderCore.
ScarRewindSystemModule.h / .cpp	Module startup and shutdown.
ScarRewindSnapshot.h	FScarRewindSnapshot struct. Edit SCAR_MAX_CUSTOM_VARS here.
ScarCircularBuffer.h	TScarCircularBuffer template. Allocation-free ring buffer.
ScarRewindTypes.h	All shared delegates, the EScarRewindState enum, and FScarRewindBookmark.
ScarRewindComponent.h / .cpp	Main component. Per-actor rewind, ghost trail with linger timer, energy, bookmarks, axis lock, post process.
ScarRewindSubsystem.h / .cpp	World Subsystem. Batch operations, global world rewind, pickup exclusion, FTickableGameObject ticking.