Weapon Bone Bridge User Manual

Version 0.5.1 | English | 23 October 2026

Published by **Pixel Tech Soft Co., Ltd.**

Weapon Bone Bridge is an Unreal Editor plugin for transferring animated weapon and helper bones from a source character to a target character. This manual explains how to build a result, correct the body and weapon motion, and export Unreal animation assets. It replaces separate release addenda with one workflow.

Requirements and validation

Item	Requirement or status
Platform	Windows 64 bit Unreal Editor. The plugin module is Editor only.
UE 5.8	Built and tested locally with UE 5.8.3 on Win64.
UE 5.5 to 5.7	Separate source variants contain fixes for the reported build errors. Builds in these exact engines have not been verified locally.
Assets	Source and target Skeletal Meshes with valid Skeletons and editable LOD 0 data. Animation is optional for bone transfer.
Dependencies	IK Rig, Control Rig, and Mesh Modeling Toolset. A compatible CR_Mannequin_Body asset is optional for native body controls.

Install and open

1. Use the package for your engine version. Place the WeaponBoneBridge folder in your project Plugins directory, then enable **Weapon Bone Bridge** in Edit > Plugins and restart the Editor.
2. If Unreal requests a source build, use the C++ toolchain required by that engine. Do not reuse a DLL built for a different engine version.
3. Open **Tools > Weapon Bone Bridge**. Keep your character meshes, animations, and optional weapon meshes in the project Content Browser.

The plugin package contains code, configuration, its icon, and the plugin descriptor. Example characters, purchased animation packs, weapons, and the Epic Control Rig asset are acquired separately. This manual is distributed separately from the plugin.

Reading order

Start with First result, then review body retargeting and viewport corrections. Weapons, constraints, and keyframes are optional. Export and saving describe how to use the generated result in your project.

First result

Build a character and animation

1. Set **SOURCE SKM** to the mesh associated with the animation pack. Set **TARGET SKM** to the character that needs the extra bones.
2. Choose a source animation in the searchable animation picker inside the tool. The picker filters assets by the source Skeleton. Leave it empty for a bone only build.
3. Choose **AUTO | Add missing bones**, or choose **MANUAL | Choose source bones** and check the bones you need. Keep **Adapt to target proportions** enabled for characters with different proportions.
4. For a first animation build, leave **Auto IK Body** enabled and **Optional Retargeted Body Override** empty. Add weapon meshes only if you need them for inspection.
5. Press **ADD BONES + RETARGET**. Read the Build report. The normal build creates a new result mesh, Skeleton, and any requested animation results.
6. Use Play / Pause and scrub the timeline. Inspect hand grips, feet, root motion, and weapon or sheath alignment throughout the clip. Rebuild after changing transfer settings or calibration.

A mesh selection alone does not retarget the preview. A message saying Reference pose or build required means the target is not showing a current result animation.

Choose which bones to transfer

Auto compares real source bones against the target using the resolved mapping. It includes unmatched bones and required parents. Inspect this list: unrelated facial, deformation, and helper bones may also be unmatched.

Manual limits the request to checked source bones. Required missing parents are added automatically. The source tree is searchable and collapsible; matching ancestors remain visible so you can understand the hierarchy.

A source bone marked **Skeleton only** exists in the Skeleton even if the current mesh reference skeleton does not contain it. These real bones can still be selected. Virtual bones are not imported as new real bones.

Understand the result

The generated mesh is based on the target. Existing target geometry, skin weights, and reference bones are checked for integrity during transfer. Source and target inputs remain unchanged until you explicitly use Overwrite Target.

Output animation assets use the **result Skeleton**. Preview or play them on the result mesh, or on a target updated through the overwrite workflow. The original unmodified target Skeleton is not automatically interchangeable with the result.

Body retargeting and bone mapping

Automatic body setup

Auto IK Body builds IK Rigs and a Retargeter for recognized humanoid skeletons. It uses available bone names and proportions to resolve the body. This supports different character sizes, but unfamiliar names, incomplete chains, or substantially different rest poses can require mapping or a configured retargeter.

Keep **Adapt to target proportions** on when the source and target limb lengths differ. Positive finite reference scale is supported; zero, negative, or invalid scale is rejected. Automatic proportions cannot determine every intended grip or contact point.

Choose the body motion source

Priority	Setting	Use
1	Optional Retargeted Body Override	An animation already retargeted to the original target Skeleton, with the same duration as the source clip.
2	Body Retargeter	Your configured compatible IK Retargeter. It takes precedence over automatic body setup.
3	Auto IK Body	Automatic humanoid setup when no valid override or custom retargeter is supplied.
4	Allow Basic FK Body	Optional fallback when Auto IK is off and no retargeter or body override is available. It is not a contact solver.

Do not place the source animation into Optional Retargeted Body Override. That field requires an already retargeted target body clip. An incompatible override can prevent a useful comparison or be cleared by input normalization.

Resolve unfamiliar skeletons

1. In the mapping settings, add **Bone Map** entries from source bone name to an existing target bone name. Explicit entries override automatic matches. Resolve root and major limb mappings before correcting weapon offsets.
2. Check the custom IK Retargeter chain definitions, retarget root, and rest pose when an automatic setup gives poor body motion.
3. Rebuild and inspect the full clip. If necessary, use Fix Transform for calibration or Keyframe mode for changes that vary with time.

Bone Map maps the animation source to the character. **Body Rig Bone Map** is separate: it maps Control Rig bone names to target bone names for native rig controls.

IK helper bones

Auto Attach IK Helpers pins mapped helper positions to the final hands, feet, and root after grip solving. Disable it if you intentionally need the source helper offsets. Yellow helper lines are skeleton visualization; moving an IK helper alone does not necessarily move the skinned body.

Workspace and transform corrections

Choose an editing target

1. Build the result, then enable **FIX TRANSFORM**. Choose **MODEL BONES** for body corrections or **WEAPON BONES** for transferred helper and weapon bones.
2. Use the searchable target bone hierarchy, or click a visible target bone line, joint, or root point in the viewport. Source bones are displayed for comparison and are not editing targets.
3. Choose **Move** or **Rotate**. Use **Local** to align the gizmo axes with the selected bone or native rig control, or **World** to use the preview world axes.
4. Drag the Unreal transform gizmo. Inspect the correction while scrubbing. Use **Reset bone** to clear the selected calibration, or Undo to reverse a supported edit.
5. Press **ADD BONES + RETARGET** again to bake static corrections into the generated output. Export only when the result is current.

Static correction or animation key

Fix Transform with Keyframe mode off changes calibration across the clip. Use it for a consistently displaced grip, weapon pivot, or body alignment. Use Keyframe mode when a correction should vary during an attack, draw, or pose change.

Moving the weapon mesh attachment offset changes the mesh relative to its attachment point. Moving a transferred bone changes its animated transform. Choose the level that matches the problem instead of applying the same correction twice.

Make more room for the viewport

Press **Full Screen Workspace** to move the current session into a maximized window. **Restore Workspace**, or closing that window, returns the session to the original tab. Undo and Redo remain available in the expanded workspace.

Collapse Weapons / Parent Constraints, optional settings, Assets / Save / Load Setup, and Build report when they are not in use. The **Bones** switch controls skeleton visibility; **Rig Controls** controls native rig shapes.

Keep corrections predictable

Record or discard an unrecorded keyframe draft before changing the selected bone or animation. Active constraints can control the same child bone and take priority during their active interval. Check constraints if an edited child appears to snap back.

A viewport correction is an authoring operation. Review the baked animation asset as well as the live preview, particularly at contact changes and between sampled frames.

Native body Control Rig

Provide the rig asset

Weapon Bone Bridge can use a compatible **CR_Mannequin_Body** asset for native Unreal body controls. The rig asset is not bundled with the plugin. Add it from the Epic Third Person template for your engine version, or migrate it with its dependencies from a compatible project.

The default asset path is /Game/Characters/Mannequins/Rigs/CR_Mannequin_Body.CR_Mannequin_Body. If your asset is elsewhere, choose it in **Body Control Rig** under the advanced settings.

1. Assign and build the source and target meshes. Complete the pelvis, upper arm, lower arm, hand, thigh, calf, and foot mappings for both sides.
2. If target names do not match the rig names, add entries in **Body Rig Bone Map** from rig bone to target bone. This is independent of the source to target Bone Map.
3. Enable **Rig Controls**, then select a visible target control. Enable **Limb IK** for IK limb editing, or disable it for FK limb editing.
4. Drag the control with Move or Rotate. Select Local or World axes as needed. Use static calibration or Keyframe mode according to whether the correction should apply to the full clip or a particular time.

Read the control shapes

Display	Meaning
Boxes	IK limb controls.
Circles	FK limb controls.
Other native shapes	Other controls supplied by the rig. Visibility depends on the rig and the active mode.

Changing Limb IK switches the actual arm and leg FK or IK settings in the rig. The tool selects the corresponding active control when possible; otherwise it clears selection. A hidden inactive control is not the active editing target.

Scope of the rig adapter

The tool uses a private rig instance. It evaluates the current pose through Backwards Solve and applies edits through Forwards Solve with a baseline to avoid an untouched pose jump. It does not edit the original Control Rig asset or the target bind pose.

Native rig controls require a compatible humanoid mapping. They do not make an arbitrary custom rig or nonhumanoid mesh compatible. If the rig cannot initialize, use bone gizmos and the available humanoid limb fallback, or correct the rig asset and mappings.

Weapon meshes and sheath alignment

Attach a primary weapon and sheath

1. Expand **PRIMARY WEAPON + SHEATH**. Assign the sword or other primary weapon mesh and choose its source and target bone or socket.
2. Assign the sheath mesh and choose its source and target attachment. Use names that exist on your assets. Katana_weapon_01 and Katana_sheath_01 are examples from one pack, not required names.
3. Enable **Lock sword + sheath to SOURCE (rebuild)** to preserve the paired source spacing and rotation using the sheath as a shared anchor. Rebuild and inspect both drawn and sheathed motion.
4. Adjust the attachment offset for a mesh pivot mismatch. Use a bone correction for a consistently incorrect animated attachment. Rebuild after changing calibration.

While the sword and sheath are locked together, independent sword calibration is ignored. The paired solver estimates reach; inspect the grip and any transition where the hand enters, leaves, or changes contact with the weapon.

Add more weapons or attachments

Open **Weapons / Parent Constraints** and press **ADD WEAPON / ATTACHMENT** for each additional prop. Each entry has its own mesh, source and target attachment, visibility, and offset. This supports multiple weapons and helper bones instead of a fixed sword and sheath pair.

Each enabled attachment must contain **exactly one Static Mesh or Skeletal Mesh**. Assign a mesh or disable an empty entry. Skeletal prop meshes are displayed in their reference pose; this attachment feature does not solve a separate animated prop skeleton.

Typing a bone or socket name does not create that bone. First select and build the required real source bones, or choose an attachment that already exists on the result.

Correct a grip

1. Check whether the mismatch comes from body pose, bone mapping, the weapon bone, or the weapon mesh pivot.
2. Correct body retargeting first. Then use Fix Transform or native body controls for the hand, and adjust the weapon attachment or transferred bone only where needed.
3. Inspect the entire clip, including finger contact, hand changes, sheath contact, and extreme poses. Use keyframes for corrections that cannot be represented by one static offset.

The tool does not infer arbitrary finger wrapping or perform general weapon collision avoidance. Very different hand shapes and weapon handles can require manual correction.

Parent constraints and child control

Constrain an animated weapon bone

1. Build the result and select the weapon or helper child bone that should follow another target bone.
2. Press **ADD WEAPON PARENT CONSTRAINT**. Choose the target parent bone. Use **Maintain Offset** when you want to preserve the current relationship instead of snapping to the parent.
3. Set Start Time and End Time in seconds. An End Time of -1 keeps the constraint active through the end of the clip.
4. Preview the interval, check the transition, and rebuild to bake the constraint into output animation.

A typical use is attaching a sword bone to a sheath or body parent while it is stored. Choose a different interval or parent for the drawn portion. Constraint behavior is authored into the output; this is not a gameplay attachment switch.

Disable weapon bone movement

WEAPON CHILD CONTROL helps configure the selected weapon bone to follow its hand or another chosen parent. **Disable Weapon Bone Move** changes how that child is evaluated:

Option	Behavior
Checked	The selected weapon stays fixed relative to the character component. It does not follow the hand.
Unchecked	The weapon resumes following the configured parent relationship.

A weapon fixed relative to the character still moves with the character component. This is not a world space freeze. Rebuild before exporting the changed behavior.

Constraint rules

- Resolve a child and parent that exist on the result. A missing attachment name is not a valid parent relationship.
- Keep one enabled constraint per child. Cyclic parent relationships are rejected.
- An active parent constraint controls the child during its interval and takes priority over static calibration and bone animation keys for that child.
- Review timed constraints for every batch clip. The same interval in seconds may describe different actions in clips of different lengths.

When the weapon jumps

Check Maintain Offset, the parent choice, and the active time interval. If the source sword and sheath relationship is locked, also review the shared anchor. Avoid stacking an attachment offset, bone calibration, and constraint offset to correct the same error.

Animation keyframes

Record a correction

1. Build a current result, enable **Keyframe mode**, and select a target bone or supported body control.
2. Pause and scrub to the time you want to edit. Move or rotate the selected target. With Auto Keyframe off, this is an unrecorded draft.
3. Press **ADD KEYFRAME** to record the correction, or enable **AUTO KEYFRAME** to record a completed gizmo drag automatically.
4. Move to another time and repeat. Play the clip and check interpolation. Press **ADD BONES + RETARGET** to bake the recorded edits into the result before export.

Use the timeline diamonds

The timeline shows diamond markers for the selected bone. Click a diamond to select the key and its time. Press **Delete key**, Delete, or Backspace to remove the selected key. Undo can restore a supported key edit.

Use **Discard draft** to abandon an unrecorded edit. Record or discard a draft before changing the bone, animation, or timeline position. Do not treat an unrecorded preview as a saved animation key.

How corrections are evaluated

Keys store transform offsets for the selected bone and source animation. Translation and rotation interpolate between keys. Values hold before the first and after the last key. If a bone has no keys and its first edit is after time zero, an identity key is inserted at time zero to preserve the unedited starting pose.

Keyframe mode edits corrections on top of the retargeted motion. It is intended for direct body and weapon fixes, rather than a replacement for every Sequencer or Control Rig animation authoring feature.

Choose the right correction

Problem	Use
Weapon offset is wrong throughout the clip	Fix Transform with Keyframe mode off, or an attachment offset if the mesh pivot is the cause.
Hand or weapon needs a timed adjustment	Keyframe mode and one or more recorded keys.
Weapon must follow a parent for an interval	A parent constraint with a reviewed start and end time.
The edited child snaps back	Check active parent constraints and paired sword or sheath locking before adding more keys.

Export animations and batch corrections

Export the current result

1. Finish corrections, record any drafts, and rebuild. Confirm the source and result animations are matched and the result is current.
2. Press **Export Animation...** and choose a Content Browser destination such as /Game/Animations/MyCharacter. You can select or enter a valid project content folder.
3. Confirm the export, then inspect the generated Animation Sequence assets in the Content Browser. Exports use unique names rather than overwriting existing assets. Cancelling creates no export copies.

This exports **Unreal animation assets**, not FBX files. The exported animation uses the result Skeleton. Keep the associated result mesh and Skeleton, or use the matching overwritten target.

Batch fix selected source clips

1. Prepare a working profile and correct the first pose of a representative source animation. Confirm the target mapping, attachment calibration, and body retargeting.
2. Select multiple source Animation Sequence assets in the Content Browser. They must use the same source Skeleton and contain valid nonadditive animation data.
3. Press **Batch Fix + Export...**. Choose the destination folder. Leave **Reuse corrected first pose** enabled to apply the frame zero correction and static calibration across the selected clips.
4. If you need animation specific keys on matching clips, turn Reuse corrected first pose off. Review which clip each key belongs to.
5. Run the batch and read the report. Inspect every exported clip at contacts, extreme poses, and constraint transitions.

Reusing the first pose transfers calibration; it does not infer a separate ideal grip or action for every animation. Action specific correction curves are excluded in this mode. Constraint times remain seconds and must make sense for each clip.

Use the exported assets

Assign the result mesh in your character and use matching animation assets in your Animation Blueprint, montage, or other project logic. **Export Equipped Blueprint** creates an Actor with the result mesh and enabled prop components; it does not create locomotion, combat logic, or an animation state machine.

To move results to another project, use Asset Actions > Migrate with their dependencies. A failed batch or save can leave partial results: inspect the report and existing assets before retrying.

Save setup and update the target

Save a reusable setup

Expand **Assets / Save / Load Setup**. **Save Setup** creates a reusable profile containing your mesh references and settings. Choose that profile through **Load setup** to resume the workflow. A project local session is also restored when the tool is reopened.

Save the profile and relevant generated assets before closing the Editor or moving the project. A setup stores references; it does not embed the meshes, animations, weapon assets, or optional Control Rig.

Undo and Redo

Use the tool Undo and Redo buttons or the supported Editor shortcuts for profile changes, gizmo drags, and key edits. These commands operate on authoring transactions. Undo does not delete generated assets and is not the restore mechanism for a saved mesh overwrite.

Overwrite the original target

1. First build and inspect a separate result. Confirm that the current TARGET SKM is the original mesh you intend to update and that the result is current.
2. Press **Overwrite Target**. The target must be a project asset under /Game and distinct from the generated result. The tool creates a backup before applying the result to the target.
3. Inspect the updated mesh and its Skeleton dependencies in the project. Keep the backup and the generated result Skeleton while validating existing animations, Animation Blueprints, sockets, and other dependent assets.
4. Use **Restore Backup** to restore the tracked target backup if needed. Restoration can be refused if the relevant geometry has changed. Do not rely on Ctrl Z to reverse a saved overwrite.

Output locations

Setting or command	Meaning
Output Folder	Project content destination for generated mesh and related build assets.
Output Name	Simple asset name without path separators.
Animation Export Folder	Default folder used by the animation export dialog.
Open Result	Opens the generated mesh for inspection.
Build report	Status, failed validation, generated assets, and details needed to diagnose an incomplete operation.

The plugin is an Editor authoring tool. Generated meshes and animations can be used by your game, but gameplay code must not depend on this Editor module at runtime.

Advanced settings and editor API

Transfer and alignment settings

Use Custom Bone Rules retains explicitly calibrated rules rather than recomputing automatic proportions. Each rule identifies Source Bone, Source Anchor, a unique new Target Bone, Target Parent, and Target Anchor. All relationships must resolve without a cycle.

Reference Axes compensates for reference orientation differences. **Anchor Local** uses matching local axes. **Anatomical** uses matched limb directions. Inspect the result when the rest poses or local axis conventions differ substantially.

Translation Scale must be positive and finite. Transform offsets require finite values, a normalized rotation, and unit offset scale. The UI constrains translation scaling to 0.001 through 100. **Root Translation Scale** controls root translation separately and defaults to 1.

Public editor functions

Editor tools can use UWBBProfile and UWBBLibrary through Blueprint or C++. C++ callers add WeaponBoneBridge as an Editor module dependency. These functions are not a packaged game API.

Function	Input and result
OpenWindow	Opens the plugin window.
Analyze and PrepareSelection	Take a profile and return success. Resolve input or transfer selection before building.
Build	Takes a profile and returns success; writes generated result references and a report. Check for partial output on failure.
SaveProfile	Takes a profile; returns its saved copy or null.
ExportAnimations	Takes a profile and destination folder; returns success.
BuildBatch	Takes a profile, source animation array, destination folder, and ReuseFirstPose flag; returns the batch profile or null.
CreateEquippedBlueprint	Takes a profile; returns success and writes the equipped Blueprint result.
OverwriteTarget and RestoreTargetBackup	Take a profile; return success. Use the tracked backup workflow.
IsResultCurrent and CanPreviewCalibration	Take a profile and return whether the corresponding operation is valid.
NormalizeAnimationInputs	Takes a profile; returns the number of input changes.
GetTransferRules	Takes a profile and returns the resolved transfer rules.

Troubleshooting and practical limits

Symptom	Check or action
Target stays in reference pose or export is unavailable	Build the current inputs. Clear a wrong body override. Confirm the selected source clip and result Skeleton.
Source bone is absent from the mesh	Check the source Skeleton as well as the mesh. Select real Skeleton only bones; virtual bones are not imported as real bones.
Native body controls do not appear	Supply a compatible CR_Mannequin_Body with dependencies, enable Rig Controls, and complete the required limb mappings.
A bone cannot be selected in the viewport	Enable Bones and click the target skeleton. Record or discard a pending draft. The source is comparison only.
Weapon is missing or equipped Blueprint creation fails	Check bone or socket names. Each enabled entry needs exactly one mesh. Disable blank entries.
Hand grip or sheath alignment is wrong	Check body mapping and retarget pose before applying offsets. Inspect paired locking and active parent constraints. Use timed keys where needed.
Build rejects the mesh or animation	Check editable LOD 0 data, valid positive reference scale, resolved mappings, and nonadditive source Animation Sequences.
Body floats or feet drift	Check body retarget root, rest pose, root translation, and source contact. Helper visualization does not itself ground the skinned character.
Save or batch is incomplete	Read the report, check disk space and writable or unlocked packages, then inspect partial results before retrying.
Module cannot load in another UE version	Use the matching source variant and toolchain. Build it in that engine; do not copy a binary from another engine.

Practical limits

Automatic matching is intended for recognizable humanoid structures. Ambiguous names, missing chains or fingers, nonhumanoid topology, and large rest pose differences can require explicit mapping, a custom retargeter, or manual correction. No result is guaranteed for every mesh.

The tool does not perform general finger wrapping or collision avoidance. Inspect exported motion between baked samples as well as at the sampled frames. Dense paired weapon sampling can increase asset size, and helper positions can differ between baked keys and a live corrected preview.

UE 5.8.3 Win64 was validated with build and editing tests, including Manny, Survival, and Grux body rig integration. UE 5.5, 5.6, and 5.7 variants still require builds in their matching engines before verified support is claimed.

Copyright Pixel Tech Soft Co., Ltd. 2026. All rights reserved.