Constraints

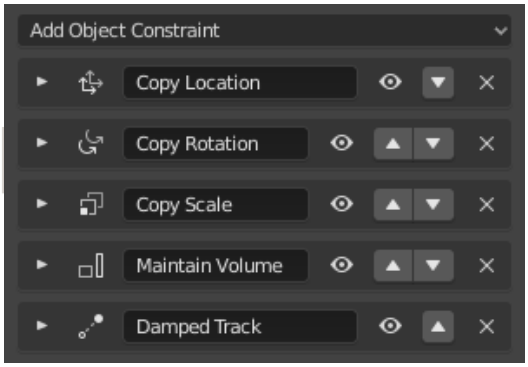
Introduction

Constraints are a way to control an object's properties (e.g. its location, rotation, scale), using either plain static values (like the "limit" ones), or another object, called "target" (like e.g. the "copy" ones).

Even though constraints are useful in static projects, their main usage is obviously in animation.

- You can control an object's animation through the targets used by its constraints (this is a form of indirect animation). Indeed, these targets can then control the constraint's owner's properties, and hence, animating the targets will indirectly animate the owner.
- You can animate constraints' settings. e.g. the Influence or when using an armature's bone as target, animate where along this bone (between root and tip) lays the real target point.

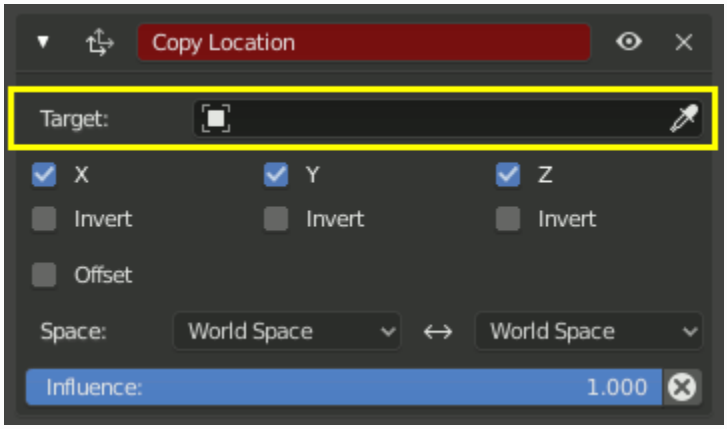
They can make the eyes of a tennis player track a tennis ball bouncing across the court, allow the wheels on a bus to all rotate together, help a dinosaur's legs bend at the knee automatically, and make it easy for a hand to grip the hilt of a sword and the sword to swing with the hand.



Common

Target

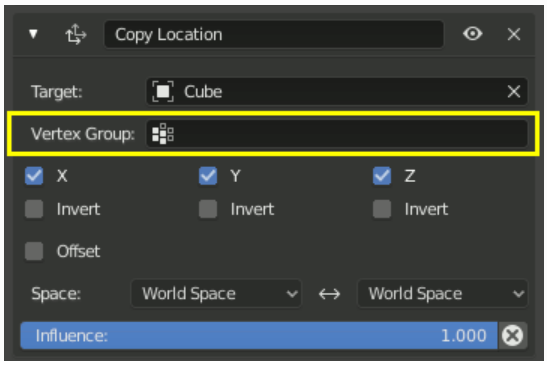
The Target Data ID field lets you link the constraint to a Target object of your choosing. This link provides data to the constraint so that it can begin to function. For example, the Copy Location Constraint needs location data to function. Fill in the Target field, and the Copy Location constraint will begin to use location data from the Target object.



The Target field must be filled in for the constraint to function.

By default, the Target will use the object origin as the target point.

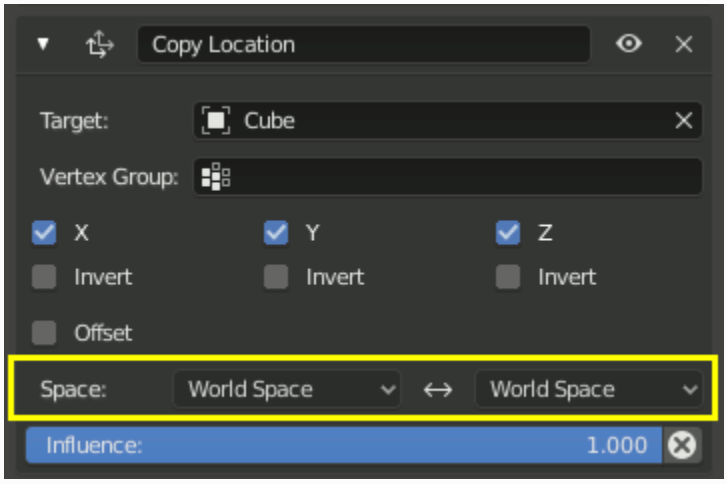
If the Target field links to a mesh or lattice object, a Vertex Group field will appear. Enter the name of a vertex group and the constraint will target the median point of this vertex group instead of the object's origin.



Space

Constraints need a frame of reference in order to function. This frame of reference is called the “space” of the constraint. Choosing one space vs. another will change this frame of reference and substantially alter the behavior of a constraint.

To understand how changing the space will change the behavior of the constraint, consider experimenting with two empties. Make sure they display as arrows so that you can see the local axes for each empty. Make sure to size one empty a little larger than the other so that they are both always visible even if directly on top of each other. Then add a constraint to one empty that targets the other and experiment thoroughly by moving, rotating and scaling the target in many different ways.



This constraint is set to use World Space as the frame of reference for both its Target Space and its Owner Space.

Target Space & Owner Space

The space used to evaluate the target of the constraint is called the Target Space. The space used to evaluate the constrained object (the object that owns the constraint) is called the Owner Space. Hover over the space select menu(s) to learn whether it affects the space of the target or the space of the owner.

When the constraints use a Target and/or/nor an Owner space there will be no, one or two selector(s). The Copy Location constraint in example use both Target **and** Owner space.

When a constraint uses both Target and Owner space, the Target and Owner can be any combination of space types.

Space Types

World Space

In this space type the world is the frame of reference for the object (or bone). Location is relative to the world origin. Rotation and Scale are oriented to the world axes. Transformations to the object, the object's parent and any other constraints higher up in the constraint stack are all taken into account.

Local Space

In this space type the parent of the object (or bone) is the frame of reference. Location is relative to the parent object origin. Rotation and Scale are oriented to the parent object axes. Only transformations to the object itself are taken into account. Transformations to the object's parent are not taken into account.

Local With Parent (bones only)

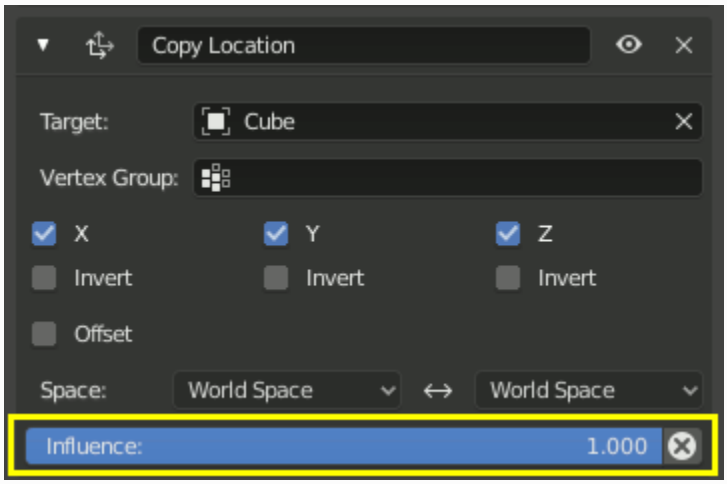
The bone properties are evaluated relative to its rest pose location and orientation, thus including both its own transformations and those caused by a possible parent relationship (i.e. the chain's transformations above the bone).

Pose Space (bones only)

The bone properties are evaluated in the armature object local space (i.e. independently from the armature transformations in Object Mode). Hence, if the armature object has null transformations, Pose Space will have the same effect as World Space.

Influence

The influence slider determines how much the constraint will affect the constrained object (target).



An influence of 0.0 will have no effect. An influence of 1.0 will have the full effect.

Values between (0.0 and 1.0) will have a partial effect, but be careful. These partial effects can be difficult to control, especially as the constraint stack grows in complexity.

Transform

Copy Location Constraint

The Copy Location constraint forces its owner to have the same location as its target.

Options

Target

Data ID used to select the constraints target, and is not functional (red state) when it has none.

X, Y, Z

These buttons control which axes are constrained.

Invert

The Invert buttons invert their respective preceding coordinates.

Offset

When enabled, this control allows the owner to be moved (using its current transform properties), relative to its target's position.

Space

Standard conversion between spaces.

Copy Rotation Constraint

The Copy Rotation constraint forces its owner to match the rotation of its target.

Options

Target

Data ID used to select the constraints target, and is not functional (red state) when it has none.

Order

Allows specifying which Euler order to use during the copy operation. Defaults to the order of the owner.

X, Y, Z

These buttons control which axes are constrained.

Invert

The Invert buttons invert their respective rotation values.

Mix

Specifies how the new rotation is combined with the existing rotation.

Replace

The new axis values replace existing values.

Add

The new axis values are added to the existing values.

Offset (Legacy)

This replicates the behavior of the original Offset checkbox. It was intended to be similar to the Before Original behavior, but does not work correctly with multiple axis rotations, and is thus deprecated.

Space

Standard conversion between spaces.

Copy Scale Constraint

The *Copy Scale* constraint forces its owner to have the same scale as its target.

Options

Target
Data ID used to select the constraints target, and is not functional (red state) when it has none.

X, Y, Z
These buttons control which axes of the target scale are copied.

Power
Allows raising the copied scale to the specified arbitrary power.

Make Uniform
Instead of copying scale for individual axes, apply a uniform scaling factor to all axes of the owner that achieves the same overall change in volume.

Offset
When enabled, the constraint combines the copied scale with the owner's scale, instead of overwriting it.

Additive
Uses addition instead of multiplication in the implementation of the Offset option.

Space
Standard conversion between spaces.

Copy Transforms Constraint

The *Copy Transforms* constraint forces its owner to have the same transforms as its target.

Options

Target

Data ID used to select the constraints target, and is not functional (red state) when it has none.

Mix

Specifies how the copied transformation is combined with the existing transformation.

Replace

The new transformation replaces the existing transformation.

Before Original

The new transformation is added before the existing transformation, as if it was applied to an imaginary parent of the constraint owner. Scale is handled like in the Aligned Inherit Scale mode of bones to avoid creating shear.

After Original

The new transformation is added after the existing transformation, as if it was applied locally to an imaginary child of the constraint owner. Scale is handled like in the Aligned Inherit Scale mode of bones to avoid creating shear.

Space

Standard conversion between spaces.

Limit Distance Constraint

The Limit Distance constraint forces its owner to stay either further from, nearer to, or exactly at a given distance from its target. In other words, the owner's location is constrained either outside, inside, or at the surface of a sphere centered on its target.

When you specify a (new) target, the Distance value is automatically set to correspond to the distance between the owner and this target.

Options

Target

Data ID used to select the constraint's target, and is not functional (red state) when it has none.

Distance

This number field sets the limit distance, i.e. the radius of the constraining sphere.

Reset Distance

When clicked, this small button will reset the Distance value, so that it corresponds to the actual distance between the owner and its target (i.e. the distance before this constraint is applied).

Clamp Region

The Limit Mode select menu allows you to choose how to use the sphere defined by the Distance setting and target's origin:

Inside

The owner is constrained inside the sphere.

Outside

The owner is constrained outside the sphere.

Surface

The owner is constrained on the surface of the sphere.

For Transform

Transform operators will take the constraint into account to immediately restrict the resulting transform property values.

Limit Location Constraint

An object or unconnected bone can be moved around the scene along the X, Y and Z axes. This constraint restricts the amount of allowed translations along each axis, through lower and upper bounds.

The limits for an object are calculated from its origin, and the limits of a bone, from its root.

It is interesting to note that even though the constraint limits the visual and rendered location of its owner, its owner's data-block still allows (by default) the object or bone to have coordinates outside the minimum and maximum ranges. This can be seen in its Transform panel.

When an owner is selected and attempted to be moved outside the limit boundaries, it will be constrained to those boundaries visually and when rendered, but internally, its coordinates will still be changed beyond the limits. If the constraint is removed, its ex-owner will seem to jump to its internally specified location.

Similarly, if its owner has an internal location that is beyond the limits, dragging it back into the limit area will appear to do nothing until the

internal coordinates are back within the limit threshold (unless you enabled the For Transform option, see below).

Setting equal the min and max values of an axis, locks the owner's movement along that axis… Although this is possible, using the Transformation Properties axis locking feature is probably easier!

Options

Minimum X, Minimum Y, Minimum Z

These buttons enable the lower boundary for the location of the owner's origin along, respectively, the X, Y and Z axes of the chosen Space. The number field below them controls the value of their limit. Note that if a min value is higher than its corresponding max value, the constraint behaves as if it had the same value as the max one.

Maximum X, Maximum Y, Maximum Z

These buttons enable the upper boundary for the location of the owner's origin along, respectively, the X, Y and Z axes of the chosen Space. Same options as above.

For Transform

We saw that by default, even though visually constrained, the owner can still have coordinates out of bounds (as shown by the Transform panel). Well, when you enable this button, this is no longer possible – the owner's transform properties are also limited by the constraint.

Limit Rotation Constraint

An object or bone can be rotated around the X, Y and Z axes. This constraint restricts the amount of allowed rotations around each axis, through lower and upper bounds.

It is interesting to note that even though the constraint limits the visual and rendered rotations of its owner, its owner's data-block still allows (by default) the object or bone to have rotation values outside the minimum and maximum ranges. This can be seen in the *Transform* panel. When an owner is rotated and attempted to be rotated outside the limit boundaries, it will be constrained to those boundaries visually and when rendered, but internally, its rotation values will still be changed beyond the limits. If the constraint is removed, its ex-owner will seem to jump to its internally specified rotation.

Similarly, if its owner has an internal rotation that is beyond the limit, rotating it back into the limit area will appear to do nothing until the internal rotation values are back within the limit threshold (unless you enabled the *For Transform* option, see below).

Setting equal the min and max values of an axis, locks the owner's rotation around that axis… Although this is possible, using the *Transformation Properties* axis locking feature is probably easier.

Limit Scale Constraint

An object or bone can be scaled along the X, Y and Z axes. This constraint restricts the amount of allowed scalings along each axis, through lower and upper bounds.

Options

Minimum/Maximum X, Y, Z

These buttons enable the lower boundary for the scale of the owner along respectively the X, Y and Z axes of the chosen Space. The Min and Max number fields to their right control the value of their lower and upper boundaries, respectively.

Action Constraint

The Action constraint is powerful. It allows you control an Action using the transformations of another object.

The underlying idea of the Action constraint is very similar to the one behind the Drivers, except that the former uses a whole action (i.e. multiple F-curves of the same type), while the latter controls a single F-curve of their “owner”…

That even if the constraint accepts the Mesh action type, only the Object, Pose and Constraint types are really working, as constraints can only affect objects’ or bones’ transform properties, and not meshes’ shapes. Also note that only the object transformation (location, rotation, scale) is affected by the action, if the action contains keyframes for other properties they are ignored, as constraints do not influence those.

As an example, let us assume you have defined an Object action (it can be assigned to any object, or even no object at all), and have mapped it on your owner through an Action constraint, so that moving the target in the (0.0 to 2.0) range along its X axis maps the action content on the

owner in the (0 to 100) frame range. This will mean that when the target's X property is 0.0 the owner will be as if in frame 0 of the linked action; with the target's X property at 1.0 the owner will be as if in frame 50 of the linked action, etc.