Character Animation Tools Documentation

General Tools

Auto-key

The auto-key feature can be found in the info bar. When it is enabled, blender will automatically set keyframes when you move objects. This is helpful for people who are not used to explicitly inserting keyframes with IKEY. There are two separate toggles for auto-keying: one for object mode and one for pose mode. These two options can be set independent of one another.

Figure 6. Auto key options

For Objects

``KeyOB" will set keyframes for objects that are moved in object mode. Users who are familiar with the blender interface will likely want to leave this option disabled.

For Actions

``KeyAC" sets keyframes for transformations done in pose mode. This ensures that you will not lose a pose by forgetting to insert keyframes. Even users who are familiar with the blender interface may find this to be a useful feature.

Ipo/Action Pinning

It is now possible to display different ipos in different windows. This is especially valuable while editing actions, which have a different ipo for each bone.

Figure 7. Pinned Action IpoWindow

You can ``pin" an ipo or action (lock it to the current window) by pressing the pin icon in the header of the window. The contents of the window will stay there, even when

the object is deselected, or another object is selected. Note that the color of the ipo block menu will change, along with the background color of the ipo window. These serve as reminders that the window is not necessarily displaying the ipo of the currently selected object.

Browsing while pinned

The browse menu is still available while a window is pinned. In this case however, changing the current data will not affect the current object; it merely changes which data is displayed.

Armature Object

Creating

A single armature will contain many bones. Consider an armature to be like a skeleton for a living creature. The arms, legs, spine and head are all part of the same skeleton object.

Figure 8. Adding an Armature

To create a new armature, select "ADD->Armature" from the toolbox. A new bone will appear with its root at the location of the 3d cursor. As you move the mouse, the bone will resize accordingly. LMB will finalize the bone and start a new one that is the child of the previous one. In this way you can make a complete chain. Pressing ESC will cancel the addition of the bone.

Adding Bones

You can add another bone to an armature while it is in edit mode by selecting "ADD-> Armature" from the toolbox again. This will start the bone-adding mode again, and the new bones you create will be a part of the current armature.

Extruding Bones

You can also extrude bones from existing bones by selecting a bone joint and pressing EKEY. The newly created bone will be a child of the bone it is extruded from.

Editing

While in edit mode, you can perform the following operations to the bones in an armature.

Adjusting

Select one or more bone joints and use any of the standard transformation operations to adjust the position or orientation of any bones in the armature. Note that IK chains

cannot have any gaps between their bones and as such moving the end point of a bone will move the start point of its child.

You can select an entire IK chain at once by moving the mouse cursor over a joint in the chain and pressing LKEY. You can also use the boundary select tool (BKEY).

Deleting

You can delete one or more bones by selecting its start and end points. When you do this you will notice the bone itself will be drawn in a highlighted color. Pressing XKEY will remove the highlighted bones. Note that selecting a single point is insufficient to delete a bone.

Point Snapping

It is possible to snap bone joints to the grid or to the cursor by using the snap menu accessible with SHIFT+S.

Numeric Mode

For more precise editing, pressing NKEY will bring up the numeric entry box. Here you can adjust the position of the start and end points as well as the bone's roll around its own axis.

Undo

While in edit mode, you can cancel the changes you have made in the current editing session by pressing UKEY. The armature will revert to the state it was in before editing began.

Joining

It is possible to join two armatures together into a single object. To do this, ensure you are in object mode, select both armatures and press CTRL+J.

Renaming

Assigning meaningful names the bones in your armatures is important for several reasons. Firstly it will make your life easier when editing actions in the action window. Secondly, the bone names are used to associate action channels with bones when you are attempting to re-use actions, and thirdly the names are used when taking advantage of the automatic pose-flipping feature.

Note that bone names need only be unique within a given armature. You can have several bone called ``Head'' so long as they are all in different armatures.

Basic Naming

To change the names of one or more bones, select the bones in edit mode and switch to the edit buttons with F9. A list of all the selected bones should appear.

Figure 9. EditButtons for an Armature

Change a bone's name by SHIFT-LMB in the bone's name box and typing a new name.

It is easier to name the bones by either only editing one bone at a time, or by making sure the ``DrawNames'' option is enabled in the EditButtons F9 (see <file:///C:/Documents%20and%20Settings/bart/My%20Documents/Disc%20toolbar/Scratchpad/Character%20Animation%20doc/PublisherDoc/PublisherDoc/html/charanim.html>).

Pose Flipping Conventions

Character armatures are typically axially symmetrical. This means that many elements are found in pairs, one on the left and one on the right. If you name them correctly, Blender can flip a given pose around the axis of symmetry, making animation of walk-cycles much easier.

For every bone that is paired, suffix the names for the left and right with either ``.L'' and ``.R'' or ``.Left'' and ``.Right''. Bones that lie along the axis of symmetry or that have no twin need no suffix. Note that the part of the name preceding the suffix should be identical for both sides. So if there are two hands, they should be named ``Hand.R'' and ``Hand.L''.

Basic Parenting

To change parenting relationships within the armature, select the bone that should be the CHILD and switch to the edit buttons window. Next to the bone there should be a menu button labeled ``Child Of''. To make the bone become the child of another bone, pick the appropriate parent from the list. Note that this is much easier if the bones have been correctly named. To dissolve a parenting relationship, choose the first (blank) entry in the list.

Note that the parenting menu only contains the names of valid parents. Bones that cannot be parents (such as children of the current bone) will not be displayed.

IK Relationship

The IK toggle next to each bone with a parent is used to determine if the IK solver should propagate its effects across this joint. If the IK button is active, the child's start point will be moved to math its parent's end point. This is to satisfy the requirement that there are no gaps in an IK chain. Deactivating the IK button will not restore the child's start point to its previous location, but moving the point will no longer affect the parent's end point.

Setting Local Axes

To get the best results while animating, it is necessary to ensure that the local axes of each bone are consistent throughout the armature. This should be done before any animation takes place.

Clearing Transforms

It is necessary that the when the armature object is in its untransformed orientation in object mode, that the front of the armature is visible in the front view, the left side is visible in the left view and so on. You can ensure this by orienting the armature so that the appropriate views are aligned and pressing CTRL+A to apply size and rotation. Again, this should be done before any animation takes place.

Adjusting Roll Handles

The orientation of the bones' roll handles is important to getting good results from the animation system.

You can adjust the roll angle of a bone by selecting it and pressing NKEY. The roll angle is the item at the bottom. The exact number that must be entered here depends on the orientation of the bone.

The Z-axis of each bone should point in a consistent direction for paired bones. A good solution is to have the Z-axes point upwards (or forwards, when the bone is vertically oriented).

This task is much easier if the ``Draw Axes" option is enabled in the edit buttons window.

Setting Weights (DEPRECIATED)

The Weight and Dist settings are only used by the automatic skinning which is a depreciated feature.

Object Mode Parenting

When making a child of an armature, several options are presented.

Parent to Bone

> In this case, the a popup menu appears allowing you to choose which bone should be the parent of the child(ren) objects.

Parent to Armature

> Choosing this option will deform the child(ren) mesh(es) according to their vertex groups. If the child meshes don't have any vertex groups, they will be subject to automatic skinning. This is very slow, so it is advised to create vertex groups instead.

Parent to Armature Object

> Choosing this option will cause the child(ren) to consider the armature to be an Empty for all intents and purposes.

Toggle Buttons for Armatures in the EditButtons F9

Figure 10. Draw options for Armatures

Rest Position Button

When this toggle is activated, the armature will be displayed in its rest position. This is useful if it becomes necessary to edit the mesh associated with an armature after some posing or animation has been done. Note that the actions and poses are still there, but they are temporarily disabled while this button is pressed.

Draw Axes Button

When this toggle is activated, the local axes of each bone will be displayed in the 3d view.

Draw Names Button

When this toggle is activated, the names of each bone will be displayed in the 3d view.

Skinning

Skinning is a technique for creating smooth mesh deformations with an armature. Essentially the skinning is the relationship between the vertices in a mesh and the bones of an armature, and how the transformations of each bone will affect the position of the mesh vertices.

Automatic (DEPRECIATED)

If a mesh does not have any vertex groups, and it is made the armature-child of an armature, Blender will attempt to calculate deformation information on the fly. This is very slow and is not recommended. It is advisable to create and use vertex groups instead.

Vertex Weights

Figure 11. Vertex Groups

Vertex groups are necessary to define which bones deform which vertices. A vertex can be a member of several groups, in which case its deformation will be a weighted average of the deformations of the bones it is assigned to. In this way it is possible to create smooth joints.

Creating

To add a new vertex group to a mesh, you must be in edit mode. Create a new vertex group by clicking on the ``New'' button in the mesh's edit buttons.

A vertex group can subsequently be deleted by clicking on the ``Delete'' button.

Change the active group by choosing one from the pull-down group menu.

Naming

Vertex groups must have the same names as the bones that will manipulate them. Both spelling and capitalization matter. Rename a vertex group by SHIFT-LMB on the name button and typing a new name. Note that vertex group names must be unique within a given mesh.

Assigning

Vertices can be assigned to the active group by selecting them and clicking the ``Assign'' button. Depending on the setting of the ``Weight'' button, the vertices will receive more or less influence from the bone. This weighting is only important for vertices that are members of more than one bone. The weight setting is not an absolute value; rather it is a relative one. For each vertex, the system calculates the sum of the weights of all of the bones that affect the vertex. The transformations of each bone are then divided by this amount meaning that each vertex always receives exactly 100% deformation.

Assigning 0 weight to a vertex will effectively remove it from the active group.

Removing

Remove vertices from the current group by selecting them and clicking the ``Remove'' button.

Selection Tools

Pressing the ``Select'' button will add the vertices assigned to the current group to the selection set. Pressing the ``Deselect'' button will remove the vertices assigned to the current group from the selection set.

Weight Painting

Weight painting is an alternate technique for assigning vertices to vertex groups. The user can ``paint'' weights onto the model and see the results in real-time. This makes smooth joints easier to achieve.

Activating

To activate weight-painting mode, select a mesh with vertex groups and click on the weight paint icon.

The active mesh will be displayed in weight-color mode. In this mode dark blue represents areas with no weight from the current group and red represent areas with full weight.

Only one group can be visualized at a time. Changing the active vertex group in the edit buttons will change the weight painting display.

Painting

Weights are painted onto the mesh using techniques similar to those used for vertex painting, with a few exceptions.

The ``color'' is the weight value specified in the mesh's edit-buttons. The ``opacity'' slider in the vertex paint buttons is used to modulate the weight.

``Erasing Weight''

To erase weight from vertices, set the weight to ``0'' and start painting.

Posemode

To manipulate the bones in an armature, you must enter pose mode. In pose mode you can only select and manipulate the bones of the active armature. Unlike edit mode, you cannot add or delete bones in pose mode.

Entering

Enter pose mode by selecting an armature and pressing CTRL+TAB. Alternatively you can activate pose mode by selecting an armature and clicking on the pose mode icon in the header of the 3d window. You can leave pose mode by the same method, or by entering edit mode.

Editing

In pose mode, you can manipulate the bones in the armature by selecting them with RMB and using the standard transformation keys: RKEY, SKEY and GKEY. Note that you cannot ``grab'' (translate) bones that are IK children of another bone.

Press IKEY to insert keyframes for selected bones.

Clearing a pose

If you want to clear the posing for one or more bones, select the bones and press ALT+R to clear rotations, ALT+S to clear scaling and ALT+G to clear translations. Issuing these three commands will all bones selected will return the armature to its rest position.

Copy/Paste/Flipped

It is frequently convenient to copy poses from one armature to another, or from one action to a different point in the same action. This is where the pose copying tools come into play.

To copy a pose, select one or more bones in pose mode, and click on the ``Copy'' button in the 3d window. The transformations of the selected bones are stored in the copy buffer until needed or until another copy operation is performed.

To paste a pose, simply click the ``Paste'' button. If ``KeyAC'' is active, keyframes will be inserted automatically.

To paste a mirrored version of the pose (if the character was leaning left in the copied pose, the mirrored pose would have the character leaning right), click on the ``Paste Flipped'' button. Note that if the armature was not set up correctly, the paste flipped technique may not work as expected.

Action Window

An action is made of one or more action channels. Each channel corresponds to one of the bones in the armature, and each channel has an Action Ipo associated with it. The action window provides a means to visualize and edit all of the ipos associated with the action.

Figure 12. ActionWindow

For every key set in a given action ipo, a marker will be displayed at the appropriate frame in the action window. This is similar to the ``Key'' mode in the ipo window.

Moving Action Keys

A block of action keys can be selected by either RMB on them or by using the boundary select tool (BKEY). Selected keys are highlighted in yellow. Once selected, the keys can be moved by pressing GKEY> and moving the mouse. Holding CTRL will lock the movement to whole-frame intervals. LMB will finalize the new location of the keys.

Scaling Action Keys

A block of action keys can be scaled horizontally (effectively speeding-up or slowing-

down the action) by selecting number of keys and pressing SKEY. Moving the mouse horizontally will scale the block. LMB will finalize the operation.

Deleting Action Keys

Delete one or more selected action keys by pressing XKEY when the mouse cursor is over the keyframe area of the action window.

Duplicating Action Keys

A block of action keys can be duplicated and moved within the same action by selecting the desired keys and pressing SHIFT+D. This will immediately enter grab mode so that the new block of keys can be moved. Subsequently LMB will finalize the location of the new keys.

Deleting Action Channels

Delete one or more entire action channels (and all associated keys) by selecting the action channels in the left-most portion of the action window (the selected channels will be highlighted in blue). With the mouse still over the left-hand portion of the window, press XKEY and confirm the deletion. Note that there is no undo so perform this operation with care.

Action IPO

The action ipo is a special ipo type that is only applicable to bones. Instead of using Euler angles to encode rotation, action ipos use quaternions, which provide better interpolation between poses.

Figure 13. ActionIpo

Quaternions

Instead of using a three-component Euler angle, quaternions use a four-component vector. It is generally difficult to describe the relationships of these quaternion channels to the resulting orientation, but it is often not necessary. It is best to generate quaternion keyframes by manipulating the bones directly, only editing the specific curves to adjust lead-in and lead-out transitions.

Action Actuator

The action actuator provides an interface for controlling action playback in the game engine. Action actuators can only be created on armature objects.

Figure 14. Action Actuator

Play Modes

Play

> Once triggered, the action will play all the way to the end, regardless of other signals is receives.

Flipper

> When it receives a positive signal, the action will play to the end. When it no longer receives a positive signal it will play from its current frame back to the start.

Loop Stop

> Once triggered, the action will loop so long as it does not receive a negative signal. When it does receive a negative signal it will stop immediately.

Loop End

> Once triggered, the action will loop so long as it does not receive a negative signal. When it does receive a negative signal it will stop only once it has reached the end of the loop.

Property

> The action will display the frame specified in the property field. Will only update when it receives a positive pulse.

Blending

By editing the ``Blendin" field you can request that Blender generates smooth transitions between actions. Blender will create a transition that lasts as many frames as the number specified in the Blendin field.

Priority

In situations where two action actuators are active on the same frame and they specify conflicting poses, the priority field can be used to resolve the conflict. The action with the lowest numbered priority will override actions with higher numbers. So priority `` 0" actions will override all others. This field is only important when two actions overlap.

Overlapping Actions

It is now possible to have two non-conflicting action actuators play simultaneously for the same object. For example, one action could specify the basic movements of the body, while a second action could be used to drive facial animation. To make this

work correctly, you should ensure that the two actions do not have any action channels in common. In the facial animation example, the body movement action should not contain channels for the eyes and mouth. The facial animation action should not contain channels for the arms and legs, etc.

Python

The following methods are available when scripting the action actuator from python.

getAction()

Returns a string containing the name of action currently associated with this actuator.

getBlendin()

Returns a floating-point number indicating the number of blending frames currently specified for this actuator.

getEnd()

Returns a floating-point number specifying the last frame of the action.

getFrame()

Returns a floating-point number indicating the current frame of playback.

getPriority()

Returns an integer specifying the current priority of this actuator.

getProperty()

Returns a string indicating the name of the property to be used for ``Property-Driven Playback".

getStart()

Returns a floating-point number specifying the first frame of the action.

setAction(action, reset)

Expects a string action specifying the name of the action to be associated with this actuator. If the action does not exist in the file, the state of the actuator is not changed.

If the optional parameter reset is set to 1, this method will reset the blending timer to 0. If the reset is set to 0 this method leaves the blending timer alone. If reset is not specified, the blending timer will be automatically reset. Calling this method does not however, change the start and end frames of the action. These may need to be set using setStart and setEnd

setBlendin(blendin)

Expects a positive floating-point number blendin specifying the number of transition frames to generate when switching to this action.

setBlendtime(blendtime)

Expects a floating-point number blendtime in the range between 0.0 and 1.0. This can

be used to directly manipulate the internal timer that is used when generating transitions. Setting a blendtime of 0.0 means that the result pose will be 100% based on the last known frame of animation. Setting a value of 1.0 means that the pose will be 100% based on the new action.

setChannel(channelname, matrix)

Accepts a string channelname specifying the name of a valid action channel or bone name, and a 4x4 matrix (a list of four lists of four floats each) specifying an overriding transformation matrix for that bone. Note that the transformations are in local bone space (i.e. the matrix is an offset from the bone's rest position).

This function will override the data contained in the action (if any) for one frame only. On the subsequent frame, the action will revert to its normal course, unless the channel name passed to setChannel is not specified in the action. If you wish to override the action for more than one frame, this method must be called on each frame.

Note that the override specified in this method will take priority over all other actuators.

setEnd(end)

Accepts a floating-point number end, which specifies what the last frame of the action should be.

setFrame(frame)

Passing a floating-point number frame allows the script to directly manipulate the actuator's current frame. This is low-level functionality for advanced use only. The preferred method is to use Property-Driven Playback mode.

setPriority(priority)

Passing an integer priority allows the script to set the priority for this actuator. Actuators with lower priority values will override actuators with higher numbers.

setProperty(propertyname)

This method accepts a string propertyname and uses it to specify the property used for Property-Driven-Playback. Note that if the actuator is not set to use Property-Playback, setting this value will have no effect.

setStart(start)

To specify the starting frame of the action, pass a floating-point number start to this method.

Constraints

Constraints are filters that are applied to the transformations of bones and objects. These constraints can provide a variety of services including tracking and IK solving.

Constraint Evaluation Rules

Constraints can be applied to objects or bones. In the case of constraints applied to bones, any constraints on the armature OBJECT will be evaluated before the

constraints on the bones are considered.

When a specific constraint is evaluated, all of its dependencies will have already been evaluated and will be in their final orientation/positions. Examples of dependencies are the object's parent, its parent's parents (if any) and the hierarchies of any targets specified in the constraint.

Within a given object, constraints are executed from top to bottom. Constraints that occur lower in the list may override the effects of constraints higher in the list. Each constraint receives as input the results of the previous constraint. The input to the first constraint in the list is the output of the ipos associated with the object.

If several constraints of the same type are specified in a contiguous block, the constraint will be evaluated ONCE for the entire block, using an average of all the targets. In this way you can constrain an object to track to the point between two other objects, for example.

Looping constraints are not allowed. If a loop is detected, all of the constraints involved will be temporarily disabled (and highlighted in red). Once the conflict has been resolved, the constraints will automatically re-activate.

Creating Objects

To add a constraint to an object, ensure you are in object mode and that the object is selected. Switch to the constraint buttons window (the icon looks like a pair of chain links) and click on the ``Add'' button.

A new constraint will appear. It can be deleted by clicking on the ``X'' icon next to it. A constraint can be collapsed by clicking on its orange triangle icon. When collapsed, a constraint can be moved up or down in the constraint list by clicking on it at choosing ``Move Up'' or ``Move Down'' from the popup menu.

For most constraints, a target must be specified in the appropriate field. In this field you must type in the name of the desired target object. If the desired target is a bone, first type in the name of the bone's armature. Another box will appear allowing you to specify the name of the bone.

Bones

To add a constraint to a bone, you must be in pose mode and have the bone selected.

Constraint Types

IK Solver

To simplify animation of multi-segmented limbs (such as arms and legs) you can add an IK solver constraint. IK constraints can only be added to bones. Once a target is specified, the solver will attempt to move the ROOT of the constraint-owning bone to the target, by re-orienting the bone's parents (but it will not move the root of the chain). If a solution is not possible, the solver will attempt to get as close as possible. Note that this constraint will override the orientations on any of the IK bone's parents.

Copy Rotation

This constraint copies the global transformation of the target and applies it to the constraint owner.

Copy Location

The constraint copies one or more axes of location from the target to the constraint owner.

Track To

This constraint causes the constraint owner to point its Y-axis towards the target. The Z-axis will be oriented according to the setting in the anim-buttons window. By default, the Z-axis will be rolled to point upwards.