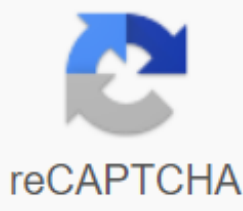




I'm not robot



Continue





Blender Simple Animation Rotation. [WML-Source: Animation.wml][TOC][Part00] Two methods are normally used in animation software to make a 3D object move. Complete positions are saved for units of time (frames). An animation is created by interpolating an object fluidly through the frames. The advantage of this method is that it allows you to work with clearly visualized units. The animator can work from one position to the next and can change previously created positions, or move them in time. Curves can be drawn for each XYZ component for location, rotation, and size. These form the graphs for the movement, with time set out horizontally and the value set out vertically. The advantage of this method is that it gives you precise control over the results of the movement. Both systems are completely integrated in Blenders "Ipo" system. Fundamentally, the Ipo system consists of standard motion curves. A simple press of a button changes the Ipo to a key system, without conversion, and with no change to the results. The user can work any way he chooses to with the keys, switching to motion curves and back again, in whatever way produces the best result or satisfies the user's preferences. This section describes the basic principles of the Ipo block, working with motion curves and the "IpoKey" system. [subsection]Ipo Block The Ipo block in Blender is universal. It makes no difference whether an object's movement is controlled or the material settings. Once you have learned to work with object Ipos, how you work with other Ipos will become obvious. Blender does distinguish between different types of Ipos. Blender concerns itself only with the type of blocks on which Ipos can work. It is better not to link object Ipos with a material as the user does not need to think about this. The interface keeps track of it automatically. Every type of Ipo block has a fixed number of available channels. These each have a name (LocX, SizeZ, etc.) that indicates how they can be applied. When you add an IpoCurve to a channel, animation begins immediately. At your discretion (and there are separate channels for this), a curve can be linked directly to a value, or it can affect a variance of this relationship. The latter enables you to move an object as per usual, with the Grabber, while the actual location is determined by IpoCurves relative to it. The Blender interface offers many options for copying Ipos, linking Ipos to more than one object (one Ipo can animate multiple objects.), or deleting Ipo links. The IpoWindow Reference section gives a detailed description of this. This chapter is restricted to the main options for application. [subsection]Making Ipos in the 3DWindow [images/InsertKeyMenu.tga] The most simple method for creating an object Ipo is with the "Insert key" ( IKEY ), command in the 3DWindow. A Pop-up menu provides a wide selection of options. We will select the topmost option: "Loc". Now the current location X-Y-Z, is saved and everything takes place automatically: If there is no Ipo block, a new one is created and linked to the object. If there are no IpoCurves in the channels "LocX", "LocY" and "LocZ", these are created. Vertices are then added in the IpoCurves with the exact values of the object location. We go 30 frames further on (3 x UPARROW ) and move the object. Again we use IKEY and immediately press ENTER . The new position is inserted in the IpoCurves. We can see this by slowly paging back through the frames (LEFTARROW ). The object moves between the two positions. In this way, you can create the animation by paging through the frames, position by position. Note that the location of the object is directly linked to the curves. When you change frames, the Ipos are always re-evaluated and re-applied. You can freely move the object within the same frame, but since you have changed frame, the object 'jumps' to the position determined by the Ipo. The rotation and size of the object are completely free in this example. They can be changed or animated with the "Insert key". [subsection]The IpoWindow [images/IPOWindow.tga] Now we want to see exactly what happened. The first Screen for this is initialised in the standard Blender start-up file. Activate this Screen with (ALT-)CTRL+LEFTARROW . At the right we see the IpoWindow displayed. This shows all the IpoCurves, the channels used and those available. You can zoom in on the IpoWindow and translate, just as everywhere else in Blender with (CTRL+MiddleMouse . In addition to the standard channels, you have the 'delta' options, such as "dLocX". These channels allow you to assign a relative change. This option is primarily used to control multiple objects with the same Ipo. In addition, it is possible to work in animation 'layers'. You can achieve subtle effects this way without having to draw complicated curves. Each curve can be selected individually with the RMB . In addition, the Grabber and Size modes operate here just as in the 3DWindow. By selecting all curves (AKEY ) and moving them to the right (GKEY ), you can move the complete animation in time. [subsection]Beziers Each curve can be placed in EditMode individually, or it can be done collectively. Select the curves and press TAB. Now the individual vertices and handles of the curve are displayed. The Bezier handles are coded, just like the curve object: Free Handle (black). This can be used any way you wish. Hotkey: HKEY (switches between Free and Aligned). Aligned Handle (pink). This arranges all the handles in a straight line. Hotkey: HKEY (toggles between Free and Aligned). Vector Handle (green). Both parts of a handle always point to the previous or next handle. Hotkey: VKEY . Auto Handle (yellow). This handle has a completely automatic length and direction. Hotkey: SHIFT+H . Handles can be moved by first selecting the middle vertex with RMB . This selects the other two vertices as well. Then immediately start the Grab mode with RMB -hold and move. Handles can be rotated by first selecting the end of one of the vertices and then use the Grabber by means of the RMB -hold and move action. As soon as handles are rotated, the type is changed automatically: Auto Handle is Aligned. Vector Handle becomes Free. "Auto" handles are placed in a curve by default. The first and last Auto handles always move horizontally, which creates a fluid interpolation. [subsection]IpoCurves The IpoCurves have an important feature that distinguishes them from normal curves: it is impossible to place more than one curve segment horizontally. Loops and circles in an Ipo are senseless and ambiguous. An Ipo can only have 1 value at a time. This is automatically detected in the IpoWindow. By moving part of the IpoCurve horizontally, you see that the selected vertices move 'through' the curve. This allows you to duplicate parts of a curve (SHIFT+D ) and to move them to another time frame. It is also important to specify how an IpoCurve must be read outside of the curve itself. There are three options for this in the IpoHeader. [images/IPOExtendModes.tga] [point]Extend mode Constant (IconBut) The ends of selected IpoCurves are continuously (horizontally) extrapolated. [point]Extend mode Direction (IconBut) The ends of the selected IpoCurves continue in the direction in which they ended. [point]Extend mode Cyclic (IconBut) The complete width of the IpoCurve is repeated cyclically. [point]Extend Mode Cyclic Extrapolation (IconBut) The complete width of the IpoCurve is extrapolated cyclic. In addition to Beziers, there are two other possible types for IpoCurves. Use the TKEY command to select them. A Pop-up menu asks what type the selected IpoCurves must be: "Constant" - after each vertex of the curve, this value remains constant. No interpolation takes place. "Linear" - linear interpolation occurs between the vertices. "Bezier" - the standard fluid interpolation. [subsection]Draw IpoCurves The IpoCurves can also be drawn 'by hand'. Use the CTRL+LMB command. Here are the rules: There is no Ipo block yet (in this window) and one channel is selected: a new new IpoBlock is created along with the first IpoCurve with one vertex. There is already an Ipo block, and a channel is selected without an IpoCurve: a new IpoCurve with one vertex is added Otherwise only a new point is added to the selected IpoCurve. This is not possible if multiple IpoCurves are selected or in EditMode. This is the best method for specifying axis rotations quickly. Select the object. In the IpoWindow, press one of the "Rot" channels and use CTRL+LMB to insert two points. If the axis rotation must be continuous, you must use the button IpoHeader->Extend mode Directional". [subsection]Rotations and Scaling One disadvantage of working with motion curves is that the freedom of transformations is limited. You can work quite intuitively with motion curves, but only if this can be processed on an XYZ basis. For a location, this is outstanding, but for a size and rotation there are better mathematical descriptions available: matrices (3x3 numbers) for size and quaternions (4 numbers) for rotation. These could also have been processed in the channels, but this can quite easily lead to confusing and mathematically complicated situations. Limiting the size to the three numbers XYZ is obvious, but this limits it to a rectangular distortion. A diagonal scaling such as 'shearing' is impossible. Simply working in hierarchies can solve this. A non-uniform scaled Parent will influence the rotation of a Child as a 'shear'. The limitation of the three number XYZ rotations is less intuitive. This so-called Euler rotation is not uniform - the same rotation can be expressed with different numbers - and has the bothersome effect that it is not possible to rotate from any position to another, the infamous gimbal lock. While working with different rotation keys, the user may suddenly be confronted with quite unexpected interpolations, or it may turn out to be impossible to force a particular axis rotation when making manual changes. Here, also, a better solution is to work with a hierarchy. A Parent will always assign the specified axis rotation to the Child. (It is handy to know that the X, Y and Z rotations are calculated one after the other. The curve that affects the "RotX" channel, always determines the X axis rotation). Fortunately, Blender calculates everything internally with matrices and quaternions. Hierarchies thus work normally, and the Rotate mode does what you would expect it to. Only the Ipos are a limitation here, but in this case the ease of use prevails above a not very intuitive mathematical purity. [subsection]IpoKeys The easiest way to work with motion curves is to convert them to IpoKeys. We return to the situation in the previous example: we have specified two positions in an object Ipo in frame 1 and frame 31 with "Insert Key". At the right of the screen, you can see an IpoWindow. We set the current frame to 21. [images/ScreenIpoKeys.tga] Press KKEY while the mouse cursor is in the 3DWindow. Two things will happen now: The IpoWindow switches to IpoKey mode. The selected object is assigned the "DrawKey" option. The two actions each have separate meanings. The IpoWindow now draws vertical lines through all the vertices of all the visible IpoCurves. Vertices with the same 'frame' value are linked to the vertical lines. The vertical lines (the "IpoKeys") can be selected, moved or duplicated, just like the vertices in EditMode. You can translate the IpoKeys only horizontally. The position of the object for each IpoKey is drawn in the 3DWindow. In addition to now being able to visualize the key positions of the object, you can also modify them in the 3DWindow. In this example, use the Grab mode on the object to change the selected IpoKeys. Below are a number of instructions for utilizing the power of the system: You can only use the RMB to select IpoKeys in the IpoWindow. Border select, and extend select, are also enabled here. Select all IpoKeys to transform the complete animation system in the 3DWindow. The 'Insert Key' always affects all selected objects. The IpoKeys for multiple objects can also be transformed simultaneously in the 3DWindow. Use the SHIFT+K command: "Show and select all keys" to transform complete animations of a group of objects all at once. Use the PAGEUP and PAGEDOWN commands to select subsequent keys in the 3DWindow. You can create IpoKeys with each arrangement of channels, by consciously excluding certain channels, you can force a situation in which changes to key positions in the 3DWindow can only be made to the values specified by the visible channels. For example, with only the channel "LocX" selected, the keys can only be moved in the X direction. Each IpoKey consists of the vertices that have exactly the same frame value. If vertices are moved manually, this can result in large numbers of keys, each having only one curve. In this case, use the JKEY ("Join") command to combine selected IpoKeys. It is also possible to assign selected IpoKeys vertices for all the visible curves: use IKEY in the IpoWindow and choose "Selected keys". The DrawKey option and the IpoKey mode can be switched on and off independently. Use the button EditButtons->DrawKey to switch off this option or object. You can switch IpoKey mode on and off yourself with KKEY in the IpoWindow. Only KKEY in the 3DWindow turns on/off both the DrawKey and IpoKey mode. [section]Animated Materials There are three ways to create an animated Material: Material Ipos. Just as with objects, IpoCurves can be used to specify 'key positions' for Materials. With the mouse in the ButtonsWindow, the command IKEY calls up a pop-up menu with options for the various Material variables. Per Material, the mapping for all 8 channels can be controlled with IpoCurves. A continuously rising curve can be placed in the channel "OfsZ", for example, to create a water-like effect in an X-Y face. Objects that give texture coordinates. Each object in Blender can be used as a source for texture coordinates. To do this, the option "Object" must be selected in the green "Coordinates input" buttons and the name of the object must be filled in. An inverse transformation is now performed on the global render coordinate to obtain the local object coordinate. This links the texture to the location, size, and rotation of the object. Animated Images. Per frame, Blender can load another (numbered) Image as a texture map. It is also possible to use SGI movie files or AVI files for this. [section]Path animation Often objects need to follow a path, or it is too hard to animate a special kind of movement with the keyframe method. Think of a planet following its way around the sun. Animating that with keyframes is virtually impossible. Curve objects can be used for the 3D display of an animation path. Only the first curve in the object is then used. Each Curve becomes a path by setting the option AnimButtons->CurvePath to ON. All Child objects of the Curve move along the specified path. It is a good idea to set the option EditButtons->3D to ON so that the paths can be freely modeled. In the ADD menu under Curve->Path, a primitive with the correct settings is available. This is a 5th order Nurbs spline, which can be used to create very fluid, continuous movements. Curves that are used as paths always get the correct number of interpolated points automatically. The button EditButtons->ResoU has no effect here. [subsection]Speed Ipo The speed along a path is determined with a curve in the IpoWindow. To see it, the Header button with the 'arrow' icon must be pressed in. The complete path runs in the IpoWindow between the vertical values 0.0 and 1.0. Drawing a curve between these values links the time to the position on the path. Backward and pulsing movements are possible with this. For most paths, an IpoCurve must run precisely between the Y-values 0.0 and 1.0. To achieve this, use the Number menu (NKEY ) in the IpoWindow. If the IpoCurve is deleted, the value of AnimButtons->PathLen determines the duration of the path. A linear movement is defined in this case. Using the option AnimButtons->CurveFollow, a rotation is also given to the Child objects of the path, so that they permanently point in the direction of the path. Use the "tracking" buttons in the AnimButtons to specify the effect of the rotation: [images/TrackButtons.tga] [point]TrackX, Y, Z -X, -Y, -Z (RowBut) This specifies the direction axis, i.e. the axis that is placed on the path. [point]UpX, UpY, UpZ (RowBut) Specifies which axis must point 'upwards', in the direction of the (local) positive Z axis. If the "Track" is the "Up" axis, it is deactivated. Curve paths cannot be given uniform rotations that are perpendicular to the local Z axis. That would make it impossible to determine the 'up' axis. To visualize these rotations precisely, we must make it possible for a Child to have its own rotations. Erase the Child's rotation with ALT+R . Also erase the "Parent Inverse": ALT+P . The best method is to 'parent' an unrotated Child to the path with the command SHIFT-CTRL+P : "Make parent without inverse". Now the Child jumps directly to the path and the Child points in the right direction. 3D paths also get an extra value for each vertex: the 'tilt'. This can be used to specify an axis rotation. Use TKEY in EditMode to change the tilt of selected vertices in EditMode, e.g. to have a Child move around as if it were on a roller coaster. [section]Timelpo With the Timelpo curve you can manipulate the animation time of objects without changing the animation or the other Ipos. In fact, it changes the mapping of animation time to global animation time. [images/TimelPO.tga] Make a simple keyframe-animation of a moving object and create a Timelpo in the IpoWindow. In frames where the slope of the Timelpo is positive, your object will advance in its animation. The speed depends on the value of the slope. A slope bigger than 1 will animate faster than the base animation. A slope smaller than 1 will animate slower. A slope of 1 means no change in the animation, negative power slopes allow you to reverse the animation. The Timelpo is especially interesting for particle systems, allowing you to "freeze" the particles or to animate particles absorbed by an object instead of emitted. Other possibilities are to make a time lapse or slow motion animation. You need to copy the Timelpo for every animation system to get a full slow motion. But by stopping only some animations, and continue to animate, for example, the camera can give some very nice effects (like those used to stunning effect the movie "The Matrix") [section]Lattice Animation You can use Lattices for two kinds of animation: Animate the vertices with vertex keys (or relative vertex keys) Move the lattice or the child object of the lattice With the second kind you can create animations that squish things between rollers, or achieve the effect of a well-known space ship accelerating to "warp"-speed. Make a space ship and add a lattice around the ship. I made the lattice with the following parameters: [images/Lattice/Lattice\_combine.tga] I put the lattice into EditMode for this picture, so you can see the vertices. For working with lattices it is also good to switch on the "Outside" option in the EditButtons for the Lattice, as this will hide the inner vertices of the lattice. Select the ship, extend the selection to the lattice (holding SHIFT while selecting), and press CTRL-P to make the lattice the parent of the ship. You should not see any deformation of the ship because the lattice is still regular. For the next few steps it is important to do them in EditMode. This causes a deformation only if the child object is inside the Lattice. So now select the lattice, enter EditMode, select all vertices (AKEY ), and scale the lattice along its x-axis (press MMB while initiating the scale) to get the stretch you want. The ship's mesh shows immediately the deformation caused by the lattice. [images/Lattice/Ship\_parented.tga] Now I edited the lattice in EditMode so that the right vertices have an increasing distance from each other. This will increase the stretch as the ship goes into the lattice. The right ends vertices I have scaled down so that they are nearly at one point; this will cause the vanishing of the ship at the end. Select the ship again and move it through the lattice to get a preview of the animation. Now you can do a normal keyframe animation to let the ship fly through the lattice. With this lattice animation, you can't use the pivot point of the object for tracking or parenting. It will move outside the object. You will need to vertex-parent an Empty to the mesh for that. To do so, select the Empty, then the mesh, enter EditMode and select one vertex, then press CTRL-P . [images/Lattice/ep\_film.tga] [section]VertexKeys VertexKeys, which should not be confused with "Object keys", the specified positions of objects, can also be created in Blender; VertexKeys are the specified positions of vertices in ObData. Since this can involve thousands of vertices, separate motion curves are not created for each vertex, but the traditional Key position system is used instead. A single IpoCurve is used to determine how interpolation is performed and the times at which a VertexKey can be seen. VertexKeys are part of ObData, not of an object. When duplicating ObData, the associated VertexKey block is also copied. It is not possible to permit multiple users to use VertexKeys in Blender, since it would not be very practical. The Key block is also universal and understands the distinction between a Mesh, a Curve, a Surface or a Lattice. Their interface and use are therefore identical. Working with Mesh VertexKeys is explained in detail in this section, which also contains a number of brief comments on the other ObData. The first VertexKey position that is created is always the reference Key. This key defines the texture coordinates. Only if this Key is active can the faces and curves, or the number of vertices, be changed. It is allowed to assign other Keys a different number of vertices. The Key system automatically interpolates this. [point]Mesh VertexKeys Creating VertexKeys in Blender is very simple, but the fact that the system is very sensitive in terms of its configuration, can cause a number of 'invisible' things to happen. The following rule must therefore be taken into consideration: As soon as a VertexKey position is inserted it is immediately active. All subsequent changes in the Mesh are linked to this Key position. It is therefore important that the Key position be added before editing begins. A practical example is given below. When working with VertexKeys, it is very handy to have an IpoWindow open. Use the first Screen from the standard Blender file, for example. In the IpoWindow, we must then specify that we want to see the VertexKeys. Do this using the Icon button with the vertex square. Go to the 3DWindow with the mouse cursor and press IKEY . With a Mesh object active, this key gives us the "Insert Key" menu with the "Mesh" option at the bottom. As soon as this has been selected, a yellow horizontal line is drawn in the IpoWindow. This is the first key and thus the reference Key. An IpoCurve is also created. Go a few frames further and again select: IKEY , ENTER (in the 3DWindow). The second Key is drawn as a light blue line. This is a normal Key; this key and all subsequent Keys affect only the vertex information. Press TAB for EditMode and translate one of the vertices in the Mesh. Then browse a few frames back: nothing happens! As long as we are in EditMode, other VertexKeys are not applied. What you see in EditMode is always the active VertexKey. Leave EditMode and browse through the frames again. We now see the effect of the VertexKey system. VertexKeys can only be selected in the IpoWindow. We always do this out of EditMode: the 'contents' of the VertexKey are now temporarily displayed in the Mesh. We can edit the specified Key by starting Editmode. There are three methods for working with Vertex Keys: The 'performance animation' method. This method works entirely in EditMode, chronologically from position to position. Insert Key. The reference is specified. A few frames further: Insert Key. Edit the Mesh for the second position. A few frames further: Insert Key. Edit the Mesh for the third position. Continue the above process... The 'editing' method. We first insert all of the required Keys, unless we have already created the Keys using the method described above. Blender is not in EditMode. Select a Key. Now start EditMode, change the Mesh and leave EditMode. Select a Key. Start EditMode, change the Mesh and leave EditMode. Continue the above process... The 'insert' method Whether or not there are already Keys and whether or not we are in EditMode does not matter in this method. Go to the frame in which the new Key must be inserted. Insert Key. Go to a new frame, Insert Key. Continue the above process... While in EditMode, the Keys cannot be switched. If the user attempts to do so, a menu appears: "Copy Key". This method can be used to copy the current key to the newly selected Key. [point]The IpoCurve and VertexKey lines Both the IpoCurve and the VertexKey lines are drawn in the IpoWindow. They can be separately selected with RMB . Since it would otherwise be too difficult working with them, selection of the Key lines is switched off when the curve is in EditMode. The channel button can be used to temporarily hide the curve (SHIFT+LMB on "Speed") to make it easier to select Keys. The Key lines in the IpoWindow can be placed at any vertical position. Select the line and use Grab mode to do this. The IpoCurve can also be processed here in the same way as described in the previous section. Instead of a 'value', however, the curve determines the interpolation between the Keys, e.g. a sine curve can be used to create a cyclical animation. With a Key line selected, three interpolation types can be specified. Press TKEY to open a menu with the options: "Linear": interpolation between the Keys is linear. The Key line is displayed as a dotted line "Cardinal": interpolation between the Keys is fluid, the standard setting. "BSpine": interpolation between the Keys is extra fluid and includes four Keys in the interpolation calculation. The positions are no longer displayed precisely, however. The Key line is drawn as a broken line. [point]Tips Key positions are always added with IKEY, even if they are located at the same position. Use this to copy positions when inserting. Two key lines at the same position can also be used to change the effect of the interpolation. If no Keys are selected, EditMode can be invoked as usual. However, when you leave EditMode, all changes are undone. Insert the Key in EditMode in this case. For Keys, there is no difference between selected and active. It is therefore not possible to select multiple Keys. When working with Keys with differing numbers of vertices, the faces can become disordered. There are no tools that can be used to specify precise sequence of vertices. This option is actually suitable only for Meshes that have only vertices such as Halos. Editbuttons->Slurp is an interesting option. The "Slurp" number indicates the interpolation of Keys per vertex with a fixed delay. The first vertex comes first and the last vertex has a delay of "Slurp" frames. This effect makes it possible to create very interesting and lively Key framing. Pay special attention to the sequence of the vertices for Meshes. They can be sorted using the command EditButtons->Xsort or made random using the command EditButons->Hash. This must of course be done before the VertexKeys are created. Otherwise, unpredictable things can will happen (this is great for Halos though!). [subsection]Curve and Surface Keys As mentioned earlier in this manual, Curve and Surface Keys work exactly the same way as Mesh Keys. For Curves, it is particularly interesting to place Curve Keys in the bevel object. Although this animation is not displayed real-time in the 3DWindow, but it will be rendered. [subsection]Lattice Keys Lattice Vertex Keys can be applied in a variety of ways by the user. When combined with "slurping", they can achieve some interesting effects. As soon as one Key is present in a Lattice, the buttons that are used to determine the resolution are blocked. [subsection]Relative VertexKeys Face by Jason Nairn Relative VertexKeys simplify the creation of facial and character animation by blending sets of VertexKeys. While traditional vertex keys are controlled with only one interpolation curve, relative vertex keys are controlled by one interpolation curve for every key position, thus relative keys can be mixed (added, subtracted, etc.). For facial animation, the base position might be a relaxed position with a slightly open mouth and eyelids half open. Then keys would be defined for left/right eye-blink, happy, sad, smiling, frowning, etc. The trick with relative vertex keys is that only the vertices that are changed between the base and the key affect the final output during blending. This means it is possible to have several keys affecting the object in different places all at the same time. For example, a face with three keys: smile, and left/right eye-blink could be animated to smile, then blink left eye, then blink right eye, then open both eyes and stop smiling - all by blending 3 keys. Without relative vertex keys 6 vertex keys would have needed to be generated, one for each target position. [point]The Relative VertexKey buttons [images/AnimButtonsRelativeVertexkeys.tga] The "Relative Keys" button (AnimButtons, F7) toggles the VertexKey system for the selected object between traditional and relative mode. It only becomes active after the first ('base') key has been inserted. [images/RelativeVertexkeysIPO1.tga] Relative keys are defined by inserting normal vertex keys. The vertical order of the vertex key determines its corresponding Ipo curve, i.e. the lowest blue key line will be controlled by the "Key1" curve, the second lowest will be controlled by the "Key2" curve, and so on. The "Relative Keys" button in the AnimButtons must be active for the Key curves to be displayed. When "Relative Keys" is active the speed curve no longer affects the mesh position and can be removed. [images/RelativeVertexkeysIPO2.tga] The Ipo curve for each key controls the blending between relative keys. These curves should be created in the typical fashion. The final position is determined by adding all of the affects of each individual Ipo curve. An important part of relative keys is the use of additive or extrapolated positions. For example, if the base position for a face is with a straight mouth, and a key is defined for a smile, then it is possible that the negative application of that key will result in a frown. Likewise, extending the Ipo Curve above 1.0 will "extrapolate" that key, making an extreme smile. [section]IKA and Skeletons by Reeavan McKay [subsection]Setting up Character Animation In this tutorial we will walk through the steps involved in preparing a character model for animation. This involves adding the bones and controls that will be used to deform the mesh. You can either use a character of your own, or you can load the "KNIGHT.BLEND" file from the CD-ROM. When designing your character, model it up so that the arms are stretched out to the sides, with the palms facing down. The legs should be straight with the feet flat on the floor (Fig. 1). Figure 1 [images/ika/notes3.tga] There are three main sets of components involved in the character animation setup process. The mesh, the skeleton, and the controls. The mesh is the actual model geometry that is rendered. The mesh is deformed by the skeleton that is composed of the IKAs and Empties that define the deformation envelope. To simplify the animation process, the skeleton is often manipulated indirectly with a series of control objects. It is a good idea to separate the different objects involved into the animation onto layers based on their functions. Consider the following layout for example: Deformed Mesh Layer Reference Mesh Layer Primary IKA skeleton Secondary IKA elements (bones that are a part of the control system but which do not directly contribute to the skeleton's deformation) Control Empties / Effector targets Deformation objects This makes it easy to show or hide layers to reveal just the mesh or the skeleton, to easily locate the control empties or to hide the deformation objects. Depending on the complexity of your skeletal system you may want more or fewer layers. Putting a character's finger IKAs on a layer of their own would it much easier to select and focus on the small bones without accidentally selecting items in the rest of the character's skeleton. [subsection]Working with IKAs [point]IK vs. FK The TAB key toggles IKA chains between Forward Kinematics (FK) and Inverse Kinematics (IK) mode. In FK mode, manipulating the IKA chain affects the root, or base of the system. In IK mode, the end or effector is controlled. Once the effector has been placed, the IKA chain will constantly try and adjust itself to that it touches the effector. You can tell which mode is active by looking at the position of the yellow dot present on the chain when it is selected. If the dot is at the base, the IKA is in FK mode. If it is at the tip, the limb is in IK mode. In a character setup, most of the major IKA chains will be left in IK mode. Note: you cannot toggle between IK mode and FK mode in the course of an animation. [point]Effector Parents Unlike other objects in Blender, IKA chains can have two parents. In addition to normal parenting relationship, an IKA can have an effector parent. An IKA chain with an effector parent will attempt to solve itself so that the tip of the chain touches the parent. If this can't be done -- if, for example the effector parent (or target) is too far away to reach -- the solver tries to get as close as possible. Note that moving the effector parent will never change the position of the start of the chain (Fig. 2). Though it is possible to directly manipulate the location of an IKA chain's effector, it is often better to use an empty as an effector parent and move that instead. This way it is possible to see where the IKA chain is trying to go, and to set targets that would not normally be reachable by an IKA chain (such as having a character's arms stretch out to touch an object that is out of reach). When making another object the parent of an IKA chain, a requestor will appear with the message EFFECTOR AS CHILD? To designate the object as an effector parent, click on the message, or press ENTER . To dismiss the requestor (and therefore set up a normal parenting relationship), press ESC or move the mouse away from the requestor. Note that effector parents only have an effect if the chain is set to IK mode. Figure 2 [images/ika/notes5.tga] [point]360 Degree Joints IKA chains in Blender are essentially two-dimensional. Though it is possible to rotate them a bit, there are usually problems when the effector target is moved out of the IKA's plane. This makes true 360 degree joints difficult to simulate. To solve this problem, we place an extra bone at the root of the main chain with the same facing, but rolled 90 degrees. This new IKA becomes the child of the main chain's root, and becomes the limb-parent to the main chain (Fig. 3). Both the chain and the limb-parent should use the same empty as an effector parent. This technique works very well for shoulder and hip joints. If you are going to be rotating an IKA operating in IK mode, you can place an empty at the root of the IKA and make it the parent of the IKA. Note that when you are building the skeleton, the joint IKAs should not be included in the calculation. Figure 3 [images/ika/notes1.tga] [point]Rolling Controls Sometimes you may find that the IKA solver leaves your limb in an awkward orientation. This happens most frequently with the arms. One way to solve this is to use an empty to control the roll of the limb. Create an empty and position it at the base of the limb (you can move the cursor to the exact location of the base by selecting the IKA and pressing SHIFT+S >>CURS->SEL. Clear the empty's rotation by pressing ALT+R >>CLEAR ROTATION. Make this empty the parent of the limb you wish to control, and then make it the child of whatever the limb was attached to. Fig. 4 shows a rolling empty applied to a 360 degree joint. When animating using a rolling control, make sure you clear the rotation before setting a new key. This will help prevent problems such as limbs that twist through impossible angles. Figure 4 [images/ika/notes6.tga] [point]Advanced Rolling Controls It is possible to roll a single limb in an IKA, rather than the entire chain, though it takes a bit more work to set it up. This is useful for joints such as elbows (The elbow is responsible for the rotation of the hand: try to rotate your palm while holding your wrist with the other hand). Start by creating an IKA chain (Fig. 5). The limb highlighted in green is the limb to which we want to add the roll. Figure 5 [images/ika/notes20.tga] Now add some single-segment IKAs: one for each limb of the arm chain. Leave these new segments in FK mode and make the appropriate limbs from the chain the limb-parents of the appropriate segments Fig. 6 shows the placement of the new bones. The blue and the green bones are parented to the main chain (represented in grey). The pink bone is left unparented. Figure 6 [images/ika/notes21.tga] Add an empty at the base of the final segment, and make sure the Z-axis of the empty points along the bone. Make the empty the limb-child of the final limb of the chain, then make the segment the child of the empty (Fig. 7). Figure 7 [images/ika/notes22.tga] When calculating the skeleton, use the three single segments, rather than the actual arm bone. Figure 8 [images/ika/notes23.tga] To rotate the empty along it's z axis, select it and press KEYPAD ' . This sets the view perpendicular to the object. Rotating the empty in this view will roll the bone along it's length. To return to the previous view, simply press the appropriate button on the keypad. [point]Root The Root is a bone that does not refer to a specific part of the model. Rather it is used to move the entire model at once, and should be the parent of all of the control empties. If you wish to scale, move or rotate your character, or to have your character move along a path for example, you only need to operate on the Root bone. The Root bone is placed at ground level between the character's feet and should be fairly large to easily distinguish it from the rest of the skeleton (Fig. 9). Figure 7 [images/ika/notes7a.tga] If you keep an un-deformed copy of the mesh on a separate layer, you can toggle the visibility of that layer to make sure that your skeleton fits the body properly. [point]Spine Depending on the design of your character, this chain may have as few as one or as many as three segments (with two segments being the typical compromise as shown in Fig. 10). More segments means more flexibility but it can make it more difficult to control the character. Figure 10 [images/ika/notes8.tga] [point]Neck/Head The neck and the head are typically two separate



IKAs as opposed to a two segment chain (Fig. 11). Normally, the neck doesn't move very much and most of the rotation actually occurs in the head. If you are building a character with real eyes, you should put a single IKA in each eye (with the root of the IKA at the center of the eyeball sphere). Figure 11 [images/ika/notes10b.tga] The head and the neck are limb-children of the last limb in the spine. The neck should be left in FK mode, while the head may be left in either FK or IK mode depending on how you intend to animate it. [point]Arms/Legs Arms and legs can be modeled with a two-segment chain, though it can be a pain to handle joint bending in such a setup. An easier method is to use a three-segment chain (Fig. 12). This allows the skin to fold properly at the joint, without suffering from severe "drinking straw" pinching artifacts. Figure 12 [images/ika/notes2.tga] The IKAs should be added from the top view so that the hinge allows the arms to bend forwards (Fig. 13). Try holding out your own arm in a similar position and watch how it bends. The arms should be limb-children of the last limb in the spine. Figure 13 [images/ika/notes13.tga] Legs can be modeled using a three-segment chain similar to the arm chain. The legs should be limb-children of the first limb in the spine (Fig. 14). Figure 14 [images/ika/notes14.tga] [point]Feet It is a good idea to use more than one bone in the feet (Fig. 15), and to make the "heel" the limb-parent of the "toe". This allows the foot to bend as the character rolls their foot forward. The last limb in the leg should be the limb-parent of the first bone in the foot. Note that the two bones have been left in FK mode to make them easier to control. Figure 15 [images/ika/notes12.tga] [point]Control Objects These are the empties that are being used as IK targets, as well as bones in FK mode that will be directly manipulated. Unless otherwise noted, empties used as controls should be limb-children of the ROOT bone. Control empties will be easier to locate and select if you click on the NAME TogBut in the EditMenu (F9 ). Two empties control the spine: The "Pelvis" empty is located at the base of the spine and acts as the parent to the spine chain. This is used to set the location of the spine. The effector of the spine is the limb-child of another empty, named "SpineTarget" (Fig. 16). Figure 16 [images/ika/notes4.tga] Hand and foot empties are used as effector parents to guide the arm and leg chains (Fig. 17). Figure 17 [images/ika/notes16.tga] [point]Deformation Objects Deformation objects are additional IKAs or empties that have been added to the skeleton in order to modify the deformation envelope. This is necessary when you find stray vertices that do not move when the rest of the limb does (Fig. 18). Figure 18 [images/ika/notes9.tga] Deformation objects are limb-parented to IKAs in the primary frame and are left in FK mode (Fig. 19). Figure 19 [images/ika/image3.tga] If you manipulate an IKA in IK mode, and then press ESC to cancel the operation, you'll probably notice that the IKA chain doesn't always revert to its base position. When you are setting up the character and you are testing out the deformation, it is a good idea to save the file before moving any limbs. When you have seen the results of the move (for better or for worse), reload the file again. This ensures that your IKAs don't fall out of alignment and saves you having to manually reposition them in their rest post. [point]Working with Skeletons Before your IKAs can smoothly deform your meshes, you must collect IKAs together and build them into a skeleton. Press CTRL+K to calculate the skeleton. Note that the highlighted bone (the bright pink one) will be the one that contains the skeleton information (Fig 20). IKAs can be used in multiple skeletons at the same time. To apply a skeleton to a mesh, make the root IKA the parent of the mesh and select the USE SKELETON option when prompted. You may notice that the mesh doesn't update immediately when you make changes to the skeleton. You can force an update by changing the frame. They easiest way to do this is to press RIGHTARROW followed by LEFTARROW . Figure 20 [images/ika/notes17.tga] You can adjust the deformation radius of the skeleton elements by adjusting the Deform Max Dist and Deform Weight entries in the root bone's edit menu accessed with F9 (Fig. 21). The Max Dist is measured in grid units and refers to the diameter of the effect (rather than the radius). Any skeleton object that has a Deform Max Dist of 0 will affect the entire mesh. This is usually not desirable, so each item in the skeleton should have a distance specified. Avoid setting the deformation distances too high or limbs may affect vertices that they normally shouldn't be able to reach. It is better to set the distance too low and to compensate with extra deformation objects. Figure 21 [images/ika/notes11.tga] The deformation zone surrounding each IKA or Empty in a skeleton is an elliptical shape (Fig. 22). This radius can be non-uniformly scaled by scaling the object, or by adjusting the Deform Max Dist settings in the skeleton's EditWindow. Figure 22 [images/ika/notes15.tga] Set the Max Dist for any empties in the skeleton to "2", and enable the "BOUNDS" TogBut in the empty's EditMenu. This makes it much easier to visualize the deformation zone of the empty. To change the size or shape of the empty's deformation, simply scale the empty and recalculate the skeleton. It is not necessary to recalculate the skeleton after making changes to the Deform Max Dist or Deform Weight entries in a skeleton although you may not see the changes until the frame changes. You can add bones to a skeleton by selecting all of the IKAs you wish to use (including all of the old ones), making sure that the root bone is highlighted, and pressing CTRL+K. Before deleting a bone, or if you wish to remove one from a skeleton, you must select all of the skeleton's IKAs except for the one you wish to remove, and recalculate the skeleton. Fig. 23 shows the spinal chain being removed. If you simply delete a bone, the skeleton's deformation weight and distance settings will be lost and you will have to manually enter them again. Figure 23 [images/ika/notes18.tga] If you change the rest position of any of the existing bones, be sure to recalculate the skeleton. Note that when the skeleton is recalculated, any meshes that use the skeleton as a parent will reset to their rest positions. Make sure you only recalculate the skeleton when it is in its rest position, or else you will find that the skeleton and the mesh are no longer in sync with each other. Depending on your character's design, it may make sense to calculate more than one skeleton. For example, the lower body and feet could use a different skeleton than the upper body. Since Blender normally only allows a single parent per object however, you must have designed your character so that the upper and lower body are separate objects. Fig. 24 shows the two skeletons, represented by different colors. The ROOT bone contains the deformation information for the lower half of the body, while the spinal chain contains the information for the upper half. The advantage to this method is that you can isolate deformations to specific parts of the body: moving the legs will not cause bending in any part of the upper body. Figure 24 [images/ika/notes19.tga] [subsection]Animating Move or rotate the entire character by manipulating the ROOT bone. You can control most of the character's limbs by simply moving the control empties and setting LOC keys for them with IKEY >>LOC. This is easiest if you hide all layers except for the ones containing the control empties and the IKA skeleton. Certain controls such as the Shoulder Roll empties may require ROT keys instead of LOC keys. The Pelvis and the ROOT will require both LOC and ROT keys. Setting LOC and ROT keys for the Pelvis empty allows you to add bounce to the character's movements, as well as twisting the torso relative to the placement of the feet. The Spine Target empty controls the amount of bend in the body itself. IKAs that have been left in FK mode (such as the head and feet) may be manipulated directly. Usually you will only need to set ROT keyframes for these limbs. Try to avoid setting keys in more IPO curves than necessary. For example, don't set ROT keys for effector targets, and don't set LOC keys for roll empties. Never set keys for any IKAs that are being controlled by the empties (which should be all of them). A good way to simplify this process is to insert LOC and/or ROT keys for all controls as appropriate on the first frame. On all subsequent frames, insert AVAIL keys. Because all objects will only receive keys in IPO curves that already exist, you will not have to decide which type of key to insert for each control. If an object doesn't have any IPO curves (such as your IKAs), they will not be affected by the INSERT command. [subsection]Conclusion The hierarchy can become quite complex as you add more controllers and bones. Fig. 25 shows an outline of all of the elements present in the demo file included on the CD. Figure 25 (DTP: Please use the chart.ai !) [images/ika/chart.tga] You can take a look at the final knight model with skeleton in the file KNIGHT\_IKA\_BLEND on the CD-ROM (fig. 26). Figure [images/ika/final.tga] [section]Camera [images/Camera.tga] Blender always renders from the active Camera. You can have an unlimited number of Cameras, but only the Camera you activate with CTRL+NUMPAD\_0 determines the display. Blender has a right-hand axis system. For a Camera display this means that the Y axis is up, the X axis runs horizontal and you look in the direction of the negative Z axis. Each object in Blender can act as a camera (with CTRL+NUMPAD\_0 ), but only a Camera block provides settings for the lens and clipping. The Spot Lamp is an exception to this. Now the "SpotSize" and shadow buffer limits are correctly displayed. Use this to precisely align a shadow Lamp and to adjust it. The command ALT+NUMPAD\_0 always return you to the last Camera object used. Each 3DWindow can have its 'own' camera, apart from that of the Scene. This camera is not rendered. Use the 'lock' IconBut in the 3DHeader for this. If it is OFF, the active camera and layer data of the 3DWindow are no longer linked to the Scene. The LocalView option of a 3DWindow can also have its own - and thus temporary - camera. Active camera's offer a number of extra options for transformations in the 3DWindow. These only work in camera view: NUMPAD\_0 : Grab mode: now there is horizontal and vertical translation, from the point of view of the camera. Use the MMB toggle to zoom in and out. Rotate mode : this allows you to align the camera with the MMB toggle. Fly mode: (SHIFT+F ). The mouse cursor jumps to the middle of the window. The operation is as follows: Mouse cursor movement determines the view direction LMB click (repeated) fly faster MMB click (repeated) fly slower LMB+MMB : sets speed to zero. CTRL : translation down (negative Z) ALT : translation up (positive Z) ESC : Camera back to the starting position; terminate Fly Mode. SPACEKEY : Leave Camera in this position. Terminate Fly Mode. (Avoid looking straight up or down. This causes irritating turbulence.) [section]Rendering The almost last step (see the next section for the last one) while working on a 3D scene is to render the stills or the animation. Depending on your needs and the media that you are producing for, you need to decide on a format of your output. For example, you may wish to make a still for printing in a high resolution and want to save it without any compression, but for a web-page you'll need to do a small JPEG-picture with high compression to cut off the loading time. [subsection]Still images The command center for all rendering are the DisplayButtons F10 . First look for the buttons defining the image size and choose here the size you need. Bear in mind, everything big costs in terms of the render time. [images/DisplayButtonsSize.tga] For quick test renderings, use the percent buttons. These will help you to render in a fraction time! Now render the picture by clicking the big "RENDER" button or by pressing the key F12 . After the rendering process is complete you can hide the render window with F11 or ESC . [images/DisplayButtonsRENDER.tga] A big impact on image quality is the "OSA" option under the "RENDER" button. Activate it to produce anti-aliased pictures. It is recommended that you use this option in most cases. You can control the value of the anti-aliasing by choosing one of the number buttons below the "OSA" button. Bigger values result in higher render times but also in better image quality. In the render window you can switch between two buffers with JKEY . This helps you comparing different render options. Now it is time to choose the output format for saving to disk. Blender offers some file types for stills that have differing capabilities. [images/DisplayButtonsFileFormats.tga] Targa: A format that saves your picture compressed without any loss. Ideal for reusing or working in 2D paint programs. Transparency information (Alpha Channel) is preserved with this format (using the RGBA option). Iris: Similar capabilities like Targa, but more common on SGI IRIX. JPEG: One of the most common formats used on the internet. But be aware that JPEG format uses a 'lossy' compression! It produces very small files, but is not recommended if you're intending to process the images at a later time. The quality setting of 100% means that there are (normally) no visible compression artifacts for the naked eye. Now hit F3 to save your image. You will get the FileWindow, where you can browse to a directory for your images and enter a filename in the lower TextButton. Now press ENTER twice to save your image. You will need to specify a filename extension if your operating system requires. [images/Filewindow\_small.tga] [subsection]Animations The way of saving animations is a little bit different than the way you save stills. In common with stills, it has a setting for the image size and for choosing an image format. When you choose an image format, Blender will save the animation as single numbered images to your hard disk. This is suitable for post-processing the pictures later. To create an animation in one part you can choose the AVI format, common for the Windows systems, or the SGI-Movie format. The AVI-format allows you to save JPEG compressed animation with all the advantages and disadvantages described above. The uncompressed AVI-format will generate big files not well suited for playing them or distribution, but ideal for processing in a computer-based video editing tool. Before we press the big "ANIM" button, we need to tell Blender where to save our animation. This is done in the text input on the left of the DisplayButtons. [images/DisplayButtonsPicsPath.tga] In the first input you can type in a valid path and the image name. You can also browse to a directory using the square button to the left of the "Pics" TextButton. Everything shown after the trailing directory separator ("/", "\" on Windows) will be used as the image name. Blender will extend the name with the number of the actual image. You can activate the "Extensions" option to let Blender append a filename extension to your files. [images/DisplayButtonsANIM.tga] Define the range that should be rendered with "Sta:" and "End:" and hit "ANIM" to start the rendering process. After Blender is ready, use "PLAY" to view the saved animation. [section]Sequence Editor Tutorial - By B@rt Veldhuizen [subsection]Introduction An often-underestimated function of Blender is the Sequence Editor. It is a complete video editing system that allows you to combine multiple video channels and add effects to them. Even though it has a limited number of operations, you can use these to create powerful video edits, especially when you combine it with the animation power of Blender! This tutorial shows you how to prepare some material and create a stunning end result. Final result [images/SeqEd/final\_result.tga] [subsection]Animation one: two cubes Let's start with something simple and see where it leads. Start a clean Blender and remove the default plane. Split the 3DWindow and switch one of the views to the camera view with NUMPAD-0 . In the top-view, add a cube and move it just outside of the dotted square that indicates the camera view. When you are planning to show your work on television, note the inner dotted square. Since not all televisions are the same, there is always a part of the picture that is 'cut off'. The inner square indicates which area is guaranteed to be viewable. The area between the dotted lines is referred to as the 'overscan area'. Moving the cube out of the camera view [images/SeqEd/cubes\_1.tga] I want to create a simple animation of the cube where it moves into view, rotates once and then disappears. Set the animation end to 61 (set the 'End:' value in the Render Buttons window - F10 ) and insert a LocRot keyframe on frame 1 with IKEY - this will store both the location and the rotation of the cube on this frame. Go to frame 21 (press ARROW\_UP twice) and move the cube closer to the camera. Insert another keyframe. On Frame 41, keep the cube on the same location but rotate it 180 degrees and insert another keyframe. Finally on frame 61 move the cube out of view, to the right and insert the last keyframe. To check, select the cube and press KKEY to show all keyframes in the 3D window. If you want, you can easily make changes by selecting a keyframe with PAGEUP or PAGEDOWN (the active keyframe will be displayed as a brighter yellow color than the other keyframes) and moving or rotating the cube. With the keys displayed, you do not need to re-insert the keyframes - they are automatically updated. Defining keyframes for the cube [images/SeqEd/cubes\_2.tga] We will need two versions of the animation: one with a solid material and one with a wireframe. For the material, I have used plain white and I have added two bright lamps - a white one and a blue one with an energy value of two.For the wireframe cube, set the material type to 'Wire' and change the color to green. A solid... [images/SeqEd/cubes\_3a.tga] ... and a wireframe cube [images/SeqEd/cubes\_3b.tga] Enter the filename in the 'Pics' field of the Render Buttons window (F10 ). Set the animation output filename. [images/SeqEd/cubes\_4a.tga] Render the animations and save them to disk as AVI files - use AVI JPG if you are short on disk space. If not, use AVI RAW for higher quality. First render the animation with the white material and save it as cube\_solid.avi. Then change the material to the green wireframe, render the animation again, and save the result as cube\_wire.avi. Press the ANIM button to start rendering. [subsection]Sequence one: delayed wireframes The first sequence will use only two wireframe animations to create an interesting effect. I will create multiple layers of video, give them a small time offset, and then add them together. This will simulate the glowing effect that you see on radar screens. Start a clean Blender file and change the 3DWindow to a Sequence Editor window by pressing SHIFT-F8 , or by selecting the Sequence Editor icon from the window header. Add a movie to the window by pressing SHIFT-A and selecting 'Movie'. From the FileSelectWindow, select the wireframe cube animation that you made before. Adding a video strip [images/SeqEd/cubes\_5.tga] After you have selected and loaded the movie file, you will see a blue strip that represents it. After adding a strip, you are automatically in grab mode. The start and end frame are now displayed in the bar. Take a closer look at the Sequence Editor screen. Depicted horizontally you can see the time value. Vertically, you can see the video 'channels'. Each channel can contain an image, a movie, or an effect. By layering different channels on top of each other, and applying effects, you can mix different sources together. If you select a video strip, its type, length, and filename will be printed at the bottom of the window. Grab your video strip and let it start at frame 1. Place it in channel 1. You can add lead-in and lead-out frames by selecting the triangles at the start and end of the strip (they will turn purple) and dragging them out. In the same way, you can define the length of a still image. Placing the strip. [images/SeqEd/cubes\_6.tga] Duplicate the movie layer with SHIFT-D , place it in channel 2 and shift it one frame to the right. You now have two layers of video on top of each other, but only one will display. To mix the two layers you need to apply an effect to them. Select both layers and press SHIFT-A . Select ADD from the requester that pops up. Mixing two video strips [images/SeqEd/cubes\_7.tga] Now split the sequence editor window and select the image button in the header. This will activate the automatic preview. If you select a frame in the sequence editor window with the strips, the preview will be automatically updated (with all the effects applied)! If you press ALT-A in the preview window, Blender will play back the animation. (Note: rendering of effects for the first time takes up a lot of processing time, so don't expect a real-time preview!). If you do not like the separate render window, switch to the Render Buttons (F10 ) and select DispView in the bottom left. Adding a preview window [images/SeqEd/cubes\_8.tga] Now it's time to add a little more mayhem to this animation! Duplicate another movie layer and add it to the ADD effect in video channel 3. Repeat this once and you will have four wireframe cubes in the preview window. All the cubes have the same brightness, but I would like to have a falloff in brightness. This is easily arranged. Open an IpoWindow somewhere (SHIFT-F6 ) and select the sequence icon in the header. Select the first add strip (the one in channel 3), hold down CTRL and LMB click in the IpoWindow on a value of 1. This sets the brightness of this add operation to maximum. Repeat this for the other two add strips, but decrease the value a bit for each of them in turn. [images/SeqEd/cubes\_9.tga] Defining the brightness of a layer with an Ipo [images/SeqEd/cubes\_10.tga] Depending on the ADD values that you have just set, your result should look something like this: Four wireframe cubes combined. [images/SeqEd/cubes\_11.tga] Now you already have 7 strips and we have only just begun on our animation, so you can imagine just how quickly the screen can become overcrowded! To make the project more manageable, select all strips (AKEY and BKEY work here, too), press MKEY , and press ENTER or click "Make Meta". The strips will now be combined into a meta-strip, and can be copied or moved as a whole. With the meta-strip selected, press N and enter a name. Here I have used 'Wire/Delay'. Named META strip [images/SeqEd/cubes\_12.tga] [subsection]Animation 2: delayed solid cubes. Now it is time to use some masks. I want to create two areas in which the animation plays back with a 1-frame time difference. This creates a very interesting glass-like visual effect. Start by creating a black and white image like this one. You can use a paint program, or you can do it in Blender. The easiest way to do this in Blender is to create a white material with an emit value of 1. In this way, you do not need to set up any lamps. Save the image as mask.tga. Animation mask [images/SeqEd/mask\_1.tga] Switch to the sequence editor and move the meta-strip (that we made before) out of the way. We will reposition it later. Add the animation of the solid cube (SHIFT+A, 'Movie'). Next, add the mask image. By default, a still image will get a length of 50 frames in the sequence editor. Change it to match the length of the cube animation by dragging out the arrows on the side of the image strip with the RMB. Now select both strips (hold down SHIFT ), press SHIFT+A , and add a SUB (subtract) effect. Subtracting the mask from the video [images/SeqEd/mask\_2.tga] In the preview window you will now see the effect; the areas where the mask is white have been removed from the picture. This effect is ready now; select all three strips and convert them into a META strip by pressing MKEY Mask subtracted [images/SeqEd/mask\_3.tga] Now do the same, except that this time you won't use the SUB effect but the MUL (multiply) effect. You will only see the original image where the mask image is white. Turn the three strips of this effect into a meta-strip again. Add the new strips next to the one from the previous steps - this makes previewing and editing much easier. You can always move them later. Mask multiplied [images/SeqEd/mask\_4.tga] For the final step I have to combine the two effects together. Move one of the meta strips above the other one and give it a time offset of one frame. Select both strips and add an ADD effect. Adding the two effects [images/SeqEd/mask\_5.tga] In the preview window, you can now see the result of the combination of the animation and the mask. When you are ready, select the two meta-strips and the ADD effect and convert them into a new meta strip. (That's right! You can have meta-strips within meta-strips!) To edit the contents of a meta-strip, select it and press TAB. The background will turn a greenish yellow to indicate that you are working inside a meta strip. Press TAB again to return to normal editing. Two time-shifted layers [images/SeqEd/mask\_6.tga] [subsection]Animation three: a tunnel To begin with, I want to have a 3D 'tunnel' animation that I can use as a background effect. This is really simple to create. First, save your current work - you will need it later. Start a new scene (CTRL-X ) and delete the default plane. Switch to front view (NUMPAD1 ). Add a 20-vertex circle about 10 units under the z=0 line (the pink line in your screen). Adding a 20-vertex circle [images/SeqEd/tunnel\_1.tga] While still in editmode, switch to side view (NUMPAD3 ) and snap the cursor to the origin by locating it roughly at the x,y,z=0 point and pressing SHIFT-S . Select 'Curs->Grid'. I want to turn the circle into a circular tube. For this, I will use the 'Spin' function. Go to the Edit Buttons window (F9) and enter a value of 180 in the 'Degr' field, and enter '10' in the 'Steps' field. Pressing 'Spin' will now rotate the selected vertices around the cursor at 180 degrees and in 10 steps. Leave EditMode (TAB ). Spinning the circle around the cursor [images/SeqEd/tunnel\_2.tga] With the default settings, Blender will always rotate and scale around the object's center, displayed as a tiny dot. This dot is yellow when the object is unselected and pink when it is selected. With the cursor still in the origin, press the 'Center Cursor' button in the Edit Buttons window to move the object center to the current cursor location. Press RKEY and rotate the tube 180 degrees around the cursor. Now it's time to move the camera into the tunnel. Open another 3DWindow and switch it to the camera view (NUMPAD+0). Position the camera in the side view window to match the screenshot- the camera view will be automatically updated. If not all of the edges of the tunnel are showing, you can force Blender to draw them by selecting 'All Edges' in the Edit Buttons window (F9 ). [images/SeqEd/tunnel\_3.tga] Moving the camera into the tunnel [images/SeqEd/tunnel\_4.tga] To make things a little easier, I want to render this as a looping animation. I can then add as many copies of it as I wish to the final video compilation. There are two things to keep in mind when creating looping animations. Make sure that there is no 'jump' in your animation when it loops. For this, you have to be careful when creating the keyframes and when setting the animation length. Create two keyframes: one with the current rotation of the tube on frame 1, and one with a rotation of 90 degrees (hold down CTRL while rotating!) on frame 51. In your animation, frame 51 is now the same as frame 1, so when rendering you will need to leave out frame 51 and render from 1 to 50. To get a linear motion, you need to remove the ease-in and ease-out of the rotation. These can be seen in the Ipo window of the tube after inserting the rotation keyframes. Select the rotation curve, enter EditMode and select all vertices and press VKEY ('Vector') to change the curve into a linear one. To create a more dramatic effect, select the camera while in camera view mode. The camera itself is displayed as the solid square. Press RKEY and rotate it a bit. If you now play back your animation it should loop seamlessly. Rotate the camera to get a more dramatic effect [images/SeqEd/tunnel\_6.tga] For the final touch, add a blue wireframe material to the tube and add a small lamp on the location of the camera. By tweaking the lamp's 'Dist' value (attenuation distance) you can make the end of the tube disappear in the dark without having to work with mist. When you are satisfied with the result, render your animation and save it as 'tunnel.avi'. A groovy tunnel : [images/SeqEd/tunnel\_7.tga] [subsection]Using the tunnel as a backdrop Reload your video compilation Blender file. The tunnel that we made in the last step will be used as a backdrop for the entire animation. To make it more interesting I will modify an ADD effect to change the tunnel into a pulsating backdrop. Prepare a completely black picture and call it 'black.tga' (try pressing F12 in an empty Blender file. Save with F3 , but make sure that you have selected the TGA file format in the Render Buttons window). Add both black.tga and the tunnel animation and combine them with an ADD effect. Setting up the backdrop effect. [images/SeqEd/tunnel\_8.tga] Now with the ADD effect selected, open an IpoWindow and select the Sequence Editor button in its header. From frame 1-50, draw an irregular line by using CTRL-LMB . Make sure that the values are between 0 and 1. When you are ready, take a look at the result in a preview screen and change the animation into a meta-strip. Save your work. Adding randomness with a irregular Ipo [images/SeqEd/tunnel\_9.tga] [subsection]Animation four: jumping logo I love randomness and chaos, so let's create some more! Take a logo (I made one just by adding a text object) and let's make it jump through the screen. Again, the easiest way to do this is to add vertices directly into the IpoWindow (select a LocX, LocY or LocZ channel first), but this time you may need to be a bit more careful with the minimum and maximum values for each channel. Don't worry about the looks of this one too much - the next step will make it hardly recognizable anyway :-)) Save the animation as 'jumpylogo.avi' . Jumping logo [images/SeqEd/logo\_1.tga] Animation five: particle bars Our last effect will use an animated mask. By combining this with the logo of the previous step, I will achieve a streaking effect that introduces the logo to our animation. This mask is made by using a particle system. To set one up, start a new Blender, switch to side view, add a plane to your scene and, while it is still selected, switch to the Animation Buttons window (F7 ). Select 'New effect' and then change the default effect (build) to 'Particles'. Change the system's settings as indicated in the figure. Press TAB to enter EditMode, select all vertices and subdivide the plane twice by pressing WKEY and selecting 'Subdivide' from the requester. Particle system settings. [images/SeqEd/particles\_1a.tga] Next switch to front view and add another plane. Scale it along the X-axis to turn it into a rectangle (press SKEY and move your mouse horizontally. Then click the middle mouse button to scale along the indicated axis only). Give the rectangle a white material with an emit value of one. Now you'll need to change the particles into rectangles by using the dupliverts function. Select the rectangle, then the particle emitter and parent them. Select only the emitter and in the left part of the animation buttons window, select the DupliVerts button. Each particle is now replaced by a rectangle. Duplivered rectangles [images/SeqEd/particles\_2.tga] I will now add some mist as a quick 'hack' to give each of the rectangles a different shade of gray. Go to the WorldButtons window, click on the [-] button in its header and select 'Add New'. The world settings will now appear. By default, the sky will now be rendered as a gradient between blue and black. Change the horizon colors (HoR, HoG, HoB) to black. Setting up mist [images/SeqEd/particles\_3a.tga] To activate rendering of mist, activate the Mist button in the middle of the screen. When using mist, you will have to indicate the distance from the camera that it will work. Select the camera, switch to the EditButtons, and enable 'ShowLimits'. Now switch to top view and return to the World Buttons window. Tweak the Sta: and Di: (Start, Distance) parameters so that the mist covers the complete width of the particle stream. Setting the mist parameters [images/SeqEd/particles\_4.tga] Set the animation length to 100 frames and render the animation to disk. Name the file "particles.avi" . Rendered particle rectangles. [images/SeqEd/particles\_5.tga] [subsection]Combining the logo and the particle bars By now you will know the drill: reload your compilation project file, switch to the Sequence Editor window and add both "particles.avi" and "logo.avi" to your project. Combine them together with a MUL effect. Since the logo animation is 50 frames and the particles animation is 100 frames, you'll need to duplicate the logo animation once and apply a second MUL effect to it. Use the logo animation twice [images/SeqEd/particles\_6.tga] Combine these three strips into one meta-strip. If you're feeling brave you can make a few copies and give them a small time offset, just like with the wireframe cube. [subsection]Animation 5: zooming logo If you choose to combine all of your animations completed so far, you'll get a really wild video compilation. But if this was to be your company's presentation, then perhaps you'd like to present the logo in a more recognizable way. The final part of our compilation will therefore concentrate on producing an animation of the logo that zooms in very slowly. Prepare this one and save it as "zoomlogo.avi" . Also prepare a white picture and save it as "white.tga" . I will now use the cross effect to make a rapid transition from black to white, then from white to our logo animation. Finally, a transition to black will conclude the compilation. Start off by placing black.tga in channel 1 and white.tga in channel 2. Make them both 20 frames long. Select them both and apply a cross effect. The cross will gradually change the resulting image from layer 1 to layer 2. In this case, the result will be a transition from black to white. Black-white transition. [images/SeqEd/particles\_8.tga] Next, add a duplicate of white.tga to layer 1 and place it directly to the right of black.tga. Make it about half as long as the original. Place the logo zoom animation in layer 2 and add a cross effect between the two. At this point, the animation looks like a white flash followed by the logo zoom animation. White-video transition [images/SeqEd/particles\_9.tga] The last thing that you'll need to do is to make sure that the animation will have a nice transition to black at the very end. Add a duplicate of black.tga and apply another cross effect. When you are ready, transform everything into a meta-strip. Video-black transition [images/SeqEd/particles\_10.tga] [subsection]Assembling everything created so far Now, let's add some of the compilations that we have made so far, and take a look at the work so far. The most important thing to keep in mind while you're creating your final compilation, is that when rendering your animation, the sequence editor only 'sees' the top layer of video. This means that you have to make sure that it is either a strip that is ready to be used, or it should be an effect like 'Add' that combines several underlying strips. The foundation of the compilation will be the fluctuating tunnel. Add a some duplicates of the tunnel meta-strip and place them in channel one. Combine them into one meta-strip. Do not worry about the exact length of the animation yet; you can always duplicate more tunnel strips. On top of that, place the delayed wireframe cube in channel 2. Add channel 1 to channel two and place the add effect in channel 3. Now I'd like to add the solid cube animation. Place it in channel 4, overlapping the wireframe animation in channel 2. Add it to the tunnel animation in layer one. This is where things are starting to get a little tricky; if you choose to leave it like this, the animation in channel 5 (the solid cube together with the tube) will override the animation in channel 2 (the wireframe cube) and the wireframe cube will become invisible as soon as the solid cube shows up. To solve this, add channel 3 to channel 5. You will often need to apply some extra add operations to fix missing parts of a video. This will most likely become apparent after you have rendered the final sequence. Slide the Sequence Editor window a little bit to the left and add the meta-strip with the particle/logo animation in it. Place this strip in layer 2 and place an add effect in layer 3. For some variation, duplicate the wireframe animation and combine it with the add in layer 3. Adding the particle/logo animation [images/SeqEd/composition\_3.tga] Now go to the end of the tunnel animation strip. There should be enough space to put the logo zoom animation at the end and still have some space left before it. If not, select the tunnel strip, press TAB and add a duplicate of the animation to the end. Press TAB again to leave meta edit mode. Adding the logo zoom animation [images/SeqEd/composition\_4.tga] Since I still have some space left, I'll add a copy of the solid cube animation. To get it to display correctly, you'll have to apply two add channels to it: one to combine it with the particle logo animation, and one to combine it with the logo zoom animation. Adding one last detail [images/SeqEd/composition\_5.tga] Here you can see my completed sequence. The complete sequence [images/SeqEd/composition\_6.tga] [subsection]Conclusion You are now ready to render your final video composition. To tell Blender to use the Sequence Editor information while rendering, select the 'Do Sequence' button in the Render Buttons window. After this, rendering and saving your animation works as before. After having persevered with this tutorial, I hope that you can now see that creating impressive video clips with Blender is a relatively easy task. It may take a bit of planning, but the tools are very flexible and fast to use. If you want to take Blender's abilities a step further, you can explore the sequence editor plug-in possibilities. With these you can write your own filters in C and apply them to your animation. Some great stuff, like depth blur and Gaussian blur, has already been posted on our web site. Be sure to check them out. Well that's it from me - I do hope you enjoyed your visit 'Sequence Editor Land' and I wish you lots of fun with Blender. B@rt The final result [images/SeqEd/final\_result.tga] -cw- Last modified: Mon Nov 6 15:55:15 CET 2000 I found a very simple solution by playing around. Go to the timeline. Insert a Delta Rotation Keyframe at the start. Then go to the object panel and chance the Delta Rotation values of the particular axis you wish to rotate. And then just add a new Delta Rotation keyframe. There you have a very simple animation of a rotation on a single axis. The most popular use cases for shape keys are in character facial animation and in tweaking and refining a skeletal rig. They are particularly useful for modeling organic soft parts and muscles where there is a need for more control over the resulting shape than what can ... For simple cases you won't notice any difference the 3D View or rendered output, however modifiers and constraints may depend on object transformation. ... animation curves or constraints. This tool should be used before rigging and animation. ... the rotation of the selection. This will make Blender consider the current rotation to be ...



