*** WORKING COPY ***

USER'S GUIDE

DRAFT 1999-03-04

Address: Cycore Computers
Dragarbrunnsgatan 35, 3tr
S-753 20 Uppsala
Sweden
Phone: +46-18-656560
Fax: +46-18-656566
E-mail: info@cycore.com

OVERVIEW

WHAT IS CULT3D?

Cult3D is the new multi-platform 3D rendering engine. It's a technological breakthrough that's 100% software. No additional hardware such as 3D accelerator cards is needed.

As a desktop or notebook user you can now view 3D objects in real-time, a feature formerly found only on expensive workstations. Cult3D gives all Internet users the opportunity to see high-quality 3D objects on-line on nearly all platforms.

Cult3D consists of two components:
1) A plugin, which allows users to view our objects on the web.
2) An exporter, which is a tool to convert objects created in 3D graphics programs to Cult3D's native format for Internet usage.

Cult3D is not a VRML viewer.
Cult3D was designed to overcome the many limitations of VRML, including image quality, file size and rendering speed.

The Cult3D player is optimized to run on all types of computers and configurations, even low-end computer systems with low bandwidth connections—such as a first generation Pentium PC or PowerMac with 14.4Kbps modem—without sacrificing responsiveness or quality.

With built-in compression and streaming capabilities, the Cult3D file format is small and efficient. This results in faster downloads and progressive viewing. The Cult3D file format is also a closed standard, which lets you publish your presentations and objects on the Internet without worrying that your work would be manipulated or used in the wrong context.

WHAT IS CULT3D DESIGNER?

Cult3D Designer is a software only, multi-platform, rendering engine delivering real-time interactive 3D graphics of unprecedented quality and speed. Cult3D Designer allows the creation of simple 3D scene animations to complex interactive 3D presentation with sounds and multi angle camera flybys.
With an event driven architecture, Cult3D Designer gives you the ability to set up all the criteria's of what your object can or can't do depending on what the user interaction does or doesn't do with your defined scene. Within your event configuration, you can use the traditional time-line animation to control a more complex animation sequence. By supporting both of these architectures, Cult3D gives you the ability to virtually do everything you set your mind to create.

Cult3D Designer is the tool for adding the interactivity to your ready made 3d objects. One can add sounds, html links, Java code, click-able areas, and movement constraints to the object. Using the Cult3D-exporter plugin in your favorite 3D-modeling/animation software enables you to create files, which the "Cult3D Authoring tool" can load. Cult3D Designer

does not import any other fileformats other than a Cult3D animation file (*.C3D). In the authoring tool you create your presentation and then export it the Cult3D file format (*.CO) which you use on your web page(s). This way you have a working data format (*.C3D) and a publishing format (*.CO) that can not be altered by untrusted parties.

Everything is defined by events. An event can be a special pressed key, timer or mouse button etc. To these events you can add actions like “play sound”, “play animation”, “goto url”, etc. Or if you will, you can program in Java to create even more advanced scenarios.

Cult3D Designer can take the ready-made animation in the *.C3D file format and define or add more complex sub-animation, i.e. interpolation (morph) between two different frames in that animation.

When publishing your work, you save it into a compressed binary fileformat with the extension .CO, which is ready to be used on Internet, Intranet or locally. Normally using vertice animation on an object can cost a lot on filesize, but when saving, you can use our advanced compression technology to further reduce files size of actual vertex animation data.

CULT3D DESIGNER REFERENCE

CULT3D DESIGNER USER INTERFACE

The Cult3D Designer main work area consists of six different sub-windows: Event Map, Scene Graph, Simulation Preview, Actions, Object Properties and Events.

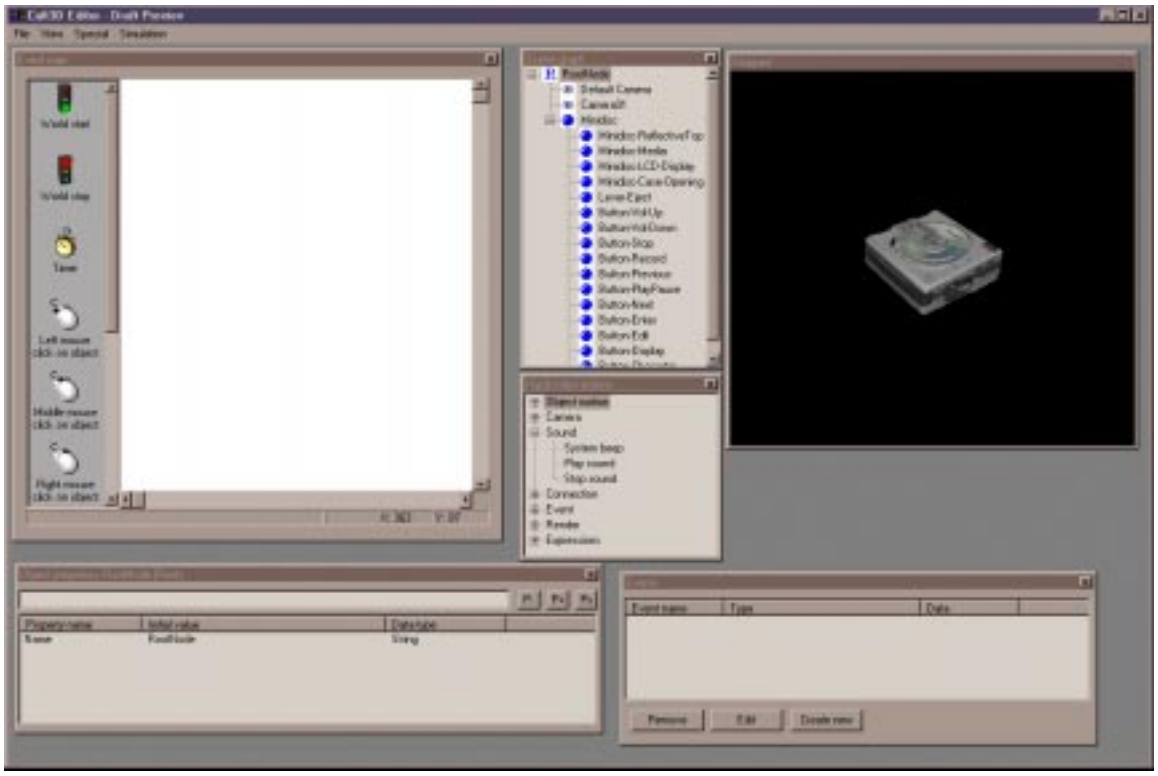


Figure 1 – Work area

Event Map

Events are a typical definition of what can happen in a scene. Every Event has an option menu, *figure 2.* Right-click with the mouse on the specified Event to access the options menu

In the options menu, you can see the different options that you can use on that particular event.

Figure 2 - Options

Initial Activation

If this option is set, it means that that event is active. If not, you will need to activate the event with an *Action* before it can be used.

Automatic Reset

If turned off, you can only use the event once. Example 1. Let say you have a "Left Mouse button Click Event" on an object in the scene. If that event doesn't have the *Automatic Reset* option checked, then the user can only press that object once for activation.

Example 2. If you have a timer event, for instance a system beep sound action attached to it, and you have set the timer preferences to 2000msec and the *Automatic Reset* option set then a beep will be played every 2 seconds.

Change Name
As the option name implies. It changes the name of the event in the Event Map.

Parameters
If the event has any user definable parameters, here is where to access that menu. You can also double click on the specific event to access the parameter dialog.

Collapse Inputs
Collapses the "tree" of objects and actions to only one icon.

Delete
Remove the Event from the Event Map.

START

The Start Event occurs when the scene begins to load.

STOP

The Stop Event occurs when one quits the scene. I.e. loads another scene.

TIMER

The Timer Event starts after a definable duration of time. By double clicking on the timer icon, you can enter a time value.

NOTE: The animation system continuosly deals with cpu breakdowns, peak conditions, system bottlenecks etc... In such a situation, timer events can't be 100% reliable. Since timer events are managed from inside the animation system pipe (which is basically a loop of functional modules), their triggering may be slightly postponed. Anyway, as all other objects, timer events are managed in the same sequence they have been created. Example: If we create two Timer Events. Timer1 and then Timer2, and we set both of them to 1000 msecs, the Timer1 will always be triggered before Timer2. The triggering code is executed at once for all the events: it can't happen that Timer2 overflows and overlaps to Timer1. Now, if we bind a Translation action to each of those two timers and connect both translation to the same object, it can happen that those two translations accumulates different relative delays during their executions. In this case, we may experience strange overlapping behaviours of the two translations. Which does not mean that the two timers behave strangely because they have the same time setting.

MOUSE BUTTON PRESSED

This Event occurs when the user is pressing the specified mouse button on the defined object to this Event. You chose between left, middle and right mouse-button for action activation.

MOUSE BUTTON RELEASED

Works like the pressed mouse button, but activates only when the user has released the specified mouse button.

Normally, the PC user has a two button mouse (left and right), and Macintosh users use a one-button mouse, but can with the “option” key, “simulate” the second mouse button. Take this into consideration when designing your events.

KEYBOARD BUTTON PRESSED

This Event occurs when the user is pressing the specified keyboard button on the defined object to this Event.

KEYBOARD BUTTON RELEASED

Works like the pressed keyboard button, but activates only when the user has released the specified button.

By double clicking on the keyboard events, one can define which button or buttons, or even a sequence of buttons should activate that specific event.

OBJECT MOTION COMPLETION

This Event occurs when the specified object motion has played its duration.

SOUND COMPLETION

This Event occurs when the specified sound or sample has played its duration.

Scene Graph
Here you can see a hierarchical view of the data from the scene. From here you select and drag’n’drop the objects you wish to work on to the Event Map.
Hierarchies of the different objects and cameras are shown here.

Preview
Everything that is done to the objects and scene is shown in this window. Here you can test your interactive presentation, to see exactly what the user will be able to do from his/her browser window. From the Simulation preview window, you can click with the left mouse

button on an object in the scene and that object is highlighted in the Scene Graph. You also have the ability to drag an object from the simulation view to the Event Map area directly by holding down the Ctrl-key while dragging the selected object. By doing this, you get a rendered icon on that specific object in the Event map which can be useful to easier understand the logic and connectivity in the Event Map.

The toolbar in the preview window has four different buttons.

The first one is for toggling “Show pivot” on/off. A pivot point is the “center” of an object, or rather the center of the rotation axis of the specific object.

The second is to reset translation/rotation to initial value. So if you have been rotating and translating an object, you press this button to reset it to the original place and angle.

The third is a toggle to be able to rotate selected object on/off. If this button isn’t pressed, then nothing happens when pressing and holding down the mouse button(s) to rotate and translate. Only selecting specific object works then.

Holding down the left mouse button = rotate
Holding down the right mouse button = zoom
Holding down the left and right mouse buttons = translate (move)
(NOTE: an object must be selected to be able to interact with it. Click once with the left mouse button to select the object, which the mouse-pointer is pointing at)

The fourth and last one is not yet implemented.
But it will be for being able to selecting more than one object to rotate (regardless of hierarchy)

These functions have nothing to do with the actual presentation. It’s only to help seeing the objects in the scene while creating the presentation. The preview doesn’t need to be run in order to be able to rotate and translate. If the third button is pressed, then you can rotate and translate selected object. It uses the hierarchy, so by selecting and rotating/translating the parent object, then the children will also rotate.

Actions
With Cult3D Designer, you get some pre-made actions. With these you can control your scene and objects to your desire. You connect an action to an event and then connect the selected data (i.e. object, sound, etc) to the action you want the data to perform.

Object motion

- *Animation play*
  If there's animation data on the selected object, this action begins the animation. By double-clicking with the mouse button on this actions icon in the Event Map, you can define different options for this action, *figure 3*.
  - *Transition time*
    Here you define how long the transition should take between the current object animation position to the next started.
  - *Motion*
    Lets the user choose between sub-motions, which are defined by right-clicking on the object in the Event Map -> Edit Motion Keyframes.
    This is good for complex animations as you can split the main motion into smaller sub-motions and assign a name to them. Then when you perform the "play motion" action, you can select which sub-motion you want to use. Let say we have an animation on a object which has both a walk and run animation of a total of 200 frames. The walk cycle is made between the 0-99 frames and the run cycle is between 100-199. Here it's quite useful to split this animation into 2 sub-motions. One called "walk animation" and the other "run animation".

Figure 3

  - *Keyframes*
    You have settings for start and end keyframes of the animation, which can be used to make a more complex animation interaction.
  - *Time*
    Defines how long the animation should last. This is by default set to the time that the original animation software set or it can be changed to either prolong or shorten the duration of the animation.
  - *Direction control*
    With the Oscillate option checked, the animation will be played forward and then backward.
  - *Iteration*
    With the "Loop" option checked, the animation is played over and over again. With the "Repeat" option checked, you have the ability to set the repeat amount to a value which you want the animation to be played.

  With the "Apply to Children" option checked, you define that all the motions shall be used on every object downwards from the hierarchy object used.

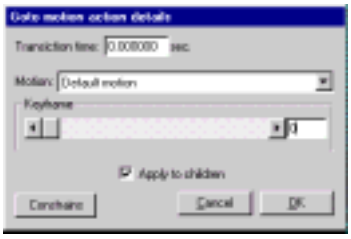


Figure 4

- *Animation 'jump to'*

Jump to a desired position in the animation. When setting a transition time greater than 0, then the object "morphs" to the new position in the animation. And the duration of the morph is defined by the transition time you wrote.

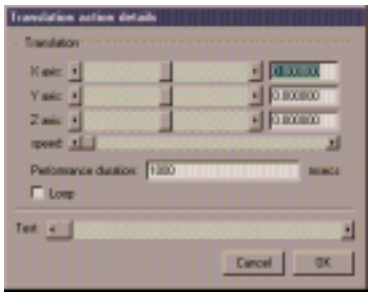


Figure 5

- *Animation stop*
  Stops the playing of the selected object

- *Translation*
  Move the selected object to a new position.
  With the sliders, you move to a new position. You can also write the actual X,Y,Z values directly for the new position. With the performance-duration, you can set how long the translation shall take. It's defined in milliseconds, so if you want the translation to the new position to take 5 seconds, then you write 5000. If Loop is checked. The translation is looped

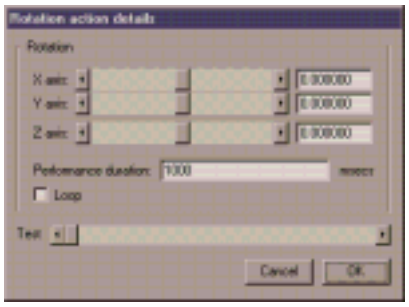


Figure 6

- *Rotation*
  Rotate the selected object. The values represent the rotation around each pivot point axis in degrees.
  You can see the options you have in *figure 6.*

Stop Translation
Stop the movements of the selected object

- *Stop Rotation*
  Stop the rotation of the selected object

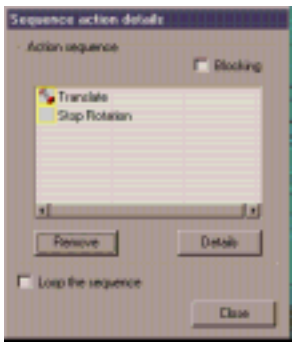


Figure 7

- *Sequence actions*
  Add a time-lined sequence of the selected object
  In this actions option menu (*figure 7*), you can see the actions connected to this sequence. If one has checked the "Blocking" option on the different actions here, one is defining to use the action in the order shown. I.e. the first action needs to be finished before the next one starts. If "Bocking" isn't checked, every action in the sequence starts directly.

- *Stop Sequence actions*
  Stop the sequence of the selected object

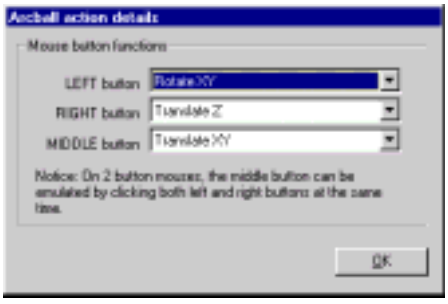


Figure 8

- *Bind arcball*
  Give the user the ability to add interaction to the connected object around its pivot point, so the user can rotate and translate the object.
  One can define the mouse button functions to that specific arcball as seen in *figure 8.*

Camera

- *Select Camera*
  Use this action to select a camera from your scene.

Figure 9

Sound

- *System Beep*
  Use this action to make a beep in the system speaker

- *Play Sound*
  Use this action to play the selected Sound resource. The options for playing a sound can be seen in *figure 9.* To add a sound resource, you'll need to access the menu->View->Sounds. The dialog in *figure 10* will be shown.

- *Stop Sound*
  Stop the playing of the selected sound resource

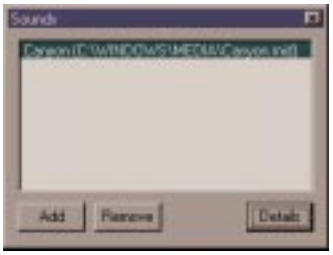


Figure 10

NOTE: Currently, it is NOT possible to play more than one sound (sample) at a time. Unless you own more than one sound board, properly installed and configured in a MCI compatible way. Since also MIDIs are played using MCI, they are subjected to all the above restrictions.
You can play a MIDI song and play a sample at the same time though.

Connection

- *Load URL*
  Double-click in this action icon to enter a desired URL destination. This causes the web browser to load the specified URL. You also have an option to specify a frame, in which to launch the new URL
- *Load Cult3D File*
  Double-click in this action icon to enter the destination to access a new Cult3D file. This replaces the current presentation in the web browser with the new Cult3D file.

Event

- *Trigger Event*
  "Signaling" means "providing the event with that signal that it is waiting for, in order to execute its bound actions". If you signal an event, this executes its actions simulating the happening of the event. An event of type "manual" gets signaled only through this action. If you have a timer event and you signal it; it's like you're forcing the timer to reach its triggering value instantaneously.

- *Reset Event*
  An event can be "reset". Resetting an event means: set its state to "not triggered" and resetting it internal counters and variables. When a timer event gets triggered, it restarts immediately counting from 0.
  Example:
  If you have assigned a left-mouse button click on an object, and then in the simulation click on that object, the event gets triggered. Then it executes its actions and reset itself, ready to detect more clicks on that object. If that event has the auto-reset flag set to false, it won't reset and it will keep signaling that the object has been clicked, until you send a

manual reset to him. In other words; until you send a reset event action to that event, you are only able to use that event once.

- *Activate Event*
  An event can be active or inactive. When an event is inactive, it doesn't work. That is: you may want to have a one-minute timer, which starts only when after clocking on a particular object. When the "left click on object" detects that you clicked on that object, you can send and "Activate Event" action to that timer, in order to make it active and starting counting to reach the triggering value of 1 minute.

- *Deactivate Event*
  This action deactivates an event. Following the previous example, you may want that your one-minute timer works only once. If you bind this action to that timer and then you drag that timer itself inside the action (the event becomes the subject of its own action), it will happen that the timer will deactivate itself when it reaches 1 minute counting (or if someone signal it manually).

- *Toggle Event Activation*
  This is a toggle for the activation/deactivation status of an event. If the event is actually active, the action deactivates it. If the event is not active, it gets activated.

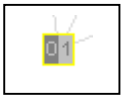
Render

- *Set background color*
  Sets the background to a new definable color
- *Hide object*
  Hides the connected object from the viewer
- *Unhide Object*
  If the object is hidden, the object becomes visible, so that the user can see it.
- *Bilinear Filter*
  Toggles if to filter all the textures in the scene with the bilinear algorithm

Expressions

- *Execute Expression*
  This action is used to "execute" the attached expression (Expressions are accessed by dragging a created expression from the Expression window to the Event Map.) The icon for an expression looks like a blackboard with crayons.

- *Conditional test*
  The "Conditional test" action's details dialog can be seen in *figure 11*. The actual icon looks like this:



It's a black and white icon with the 0 on the black area and a 1 on the white. This represents true or false. Black (0) stands for false and White (1) for true.

To this action you can connect other actions and when you drop that action onto this "conditional test" action, you can select either the true or false side of it.

Bringing up the details on the "conditional test" action gives you the ability to specify the criteria of the actual condition.

To add which property to test, you just drag the specific object property from the object property window and release it in the property edit box. Then you can specify the condition and what to compare with. You can if you want compare two different object properties. Once again, just select the specific objects property you want to compare with from the object property window and drag and drop it into the "compare with" edit box.

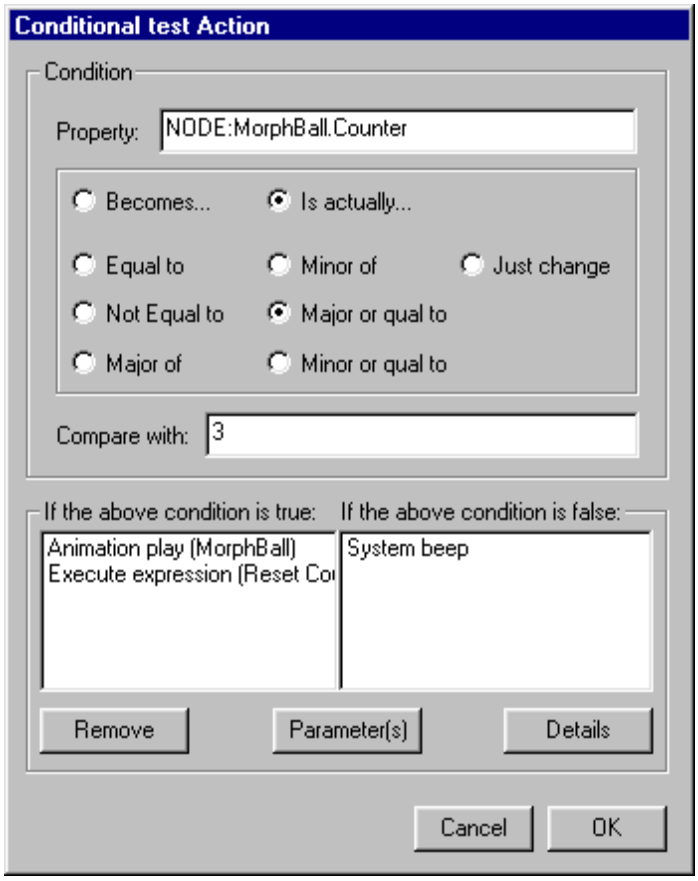


Figure 11

Object Properties

Object properties are good for to store and use data for your presentation. The properties you use or create are for each object in the scene.

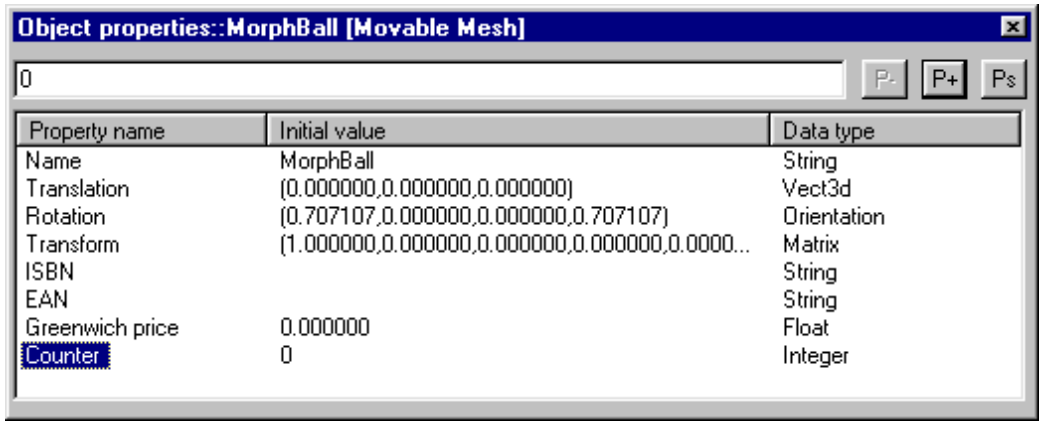


Figure 12

By selecting an object, you'll see the properties on that specific object (*figure 12)*. There are 3 columns for each property. Property name, initial value data for that property and what data type it is.

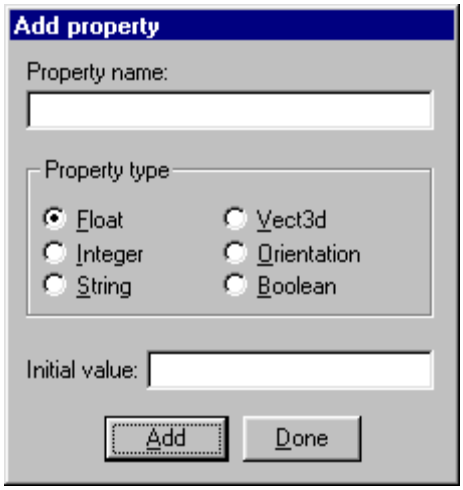


Figure 13

The initial properties for every object in the scene are the first seven properties seen in the figure. (all except the Counter property). Those seven are system given object properties. They can't be removed and is always attached to each object in the scene. A camera object only has 4 system given properties. Name, Translation, Rotation and Transform.

The Counter property is a "User Defined Object Property", which the user can create using the "P+" button. Pressing The "P+" or the "Ps" button brings up the dialog shown in *figure 13.* A unique name must be used for the new property.
NOTE: Characters such as: .(dot), -(minus), +(plus), *(product) and /(division) should NOT be used in the name.

The "P-" removes a selected user defined object property, but this function isn't implemented in this current version of c3dd though and will always be grayed out. A system given object property cannot be removed.

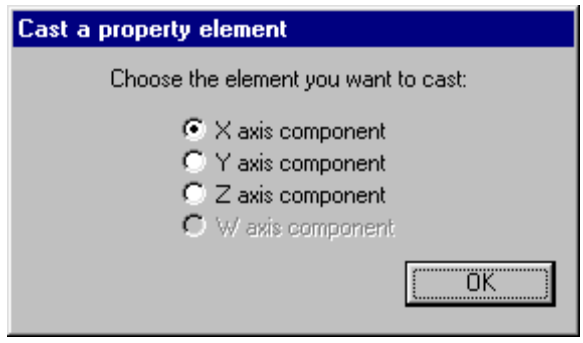


Figure 14

The "Ps" button is for creating a reference of an object property. For instance, by deriving a "Translation" or a "Rotation" property will first show the "Add property" dialog seen in *figure 13.* And after writing the name and setting the data type to either Float or Integer will bring up the dialog shown in *figure 14.*
Here you specify which axis the property should handle.
When you're done adding properties, press the "done" button.

Events

Here are all events in the scene displayed, for another view of your created scene criteria's. If you for instance want to change an event type, you can easily do it here, instead of removing the old event with all it's connections, and then create the new one and connect the old connections to the new one.

MENUS

File

Load object
Loads a Cult3D Animation File (*.C3D).

Save object
Saves the current Cult3D presentation by overwriting the last save of the presentation.
The output is a Cult3D File (*.CO).
Note: One can NOT load a .CO file into this program again, because of the "protection" of the presentation when one distribute the file, so that no one can manipulate the presentation. Check a more comprehensive description for the save dialog further down in the document.

Load project
Loads a Cult3D Project File (*.C3P)
This file includes the *.C3D file location, all the events, actions, etc and the layout of the presentation.

Save project
Saves a Cult3D Project File (*.C3P)
If no project was previously saved, this option works like Save project as…
Note: Everything is saved except the actual object.
This makes it possible to change the actual object a little without having to re-make all the layout and settings.

Save project as…
Saves a Cult3D Project File (*.C3P) under a different filename.

Preferences
Lets the user define:

- Which Java Editor to use
- Which Java compiler, and compiler options
- Default paths to: C3D Files, Java files, Resource files, Sound files, CO files and Project files.
- Event Map settings:

Exit
Exits the program

Window
Scene Graph
Toggle this window to show or not.

Object Properties
Toggle this window to show or not.

Event Map
Toggle this window to show or not.

Events
Toggle this window to show or not.

Actions
Toggle this window to show or not.

Java actions
Toggle this window to show or not.
Here one loads java classes into the project

Resources
Toggle this window to show or not.
Here one loads resource files into the project

Sounds
Toggle this window to show or not.
Here one loads the sounds into the project

Expressions
The two expressions shown in the list in *figure 15* is created and are not there by default.

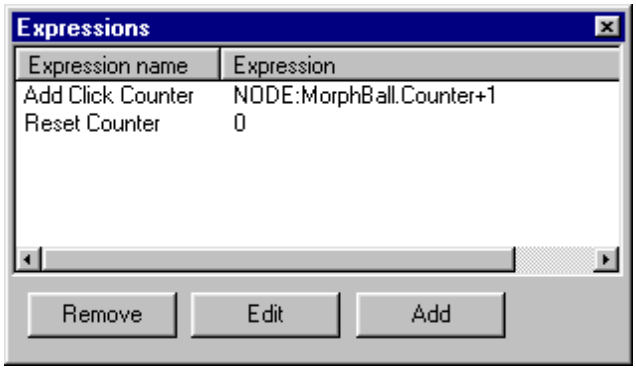


Figure 15

But if I would like to create one of them, I just click on the "add" button in this window to bring fourth the "Expression editor" dialog shown in *figure 16.*

First of I need to set a name to identify the expression I'm trying to create and in this case I write the name "Add click counter". In the editor I can access and use the system given object properties and user defined object properties. So what you can do is DRAGGING the object properties to the big white area inside the expression editor. In this case I've dragged a user defined object property for an object called "MorphBall" and the property named "Counter".
I've also added +1 to it and decided to store the new data in the same property. (just drag the property to the destination edit box too)

So, here I take the value I have from the MorphBall's Counter property and add 1 to it. So if the Counter value was 3 before executing the expression, then the value has changed to 4 after the execution. The Editor is only for specifying on what should be done, the actual execution isn't done until creating an action logically called "execute expression" and assigning that to the expression you want to execute.

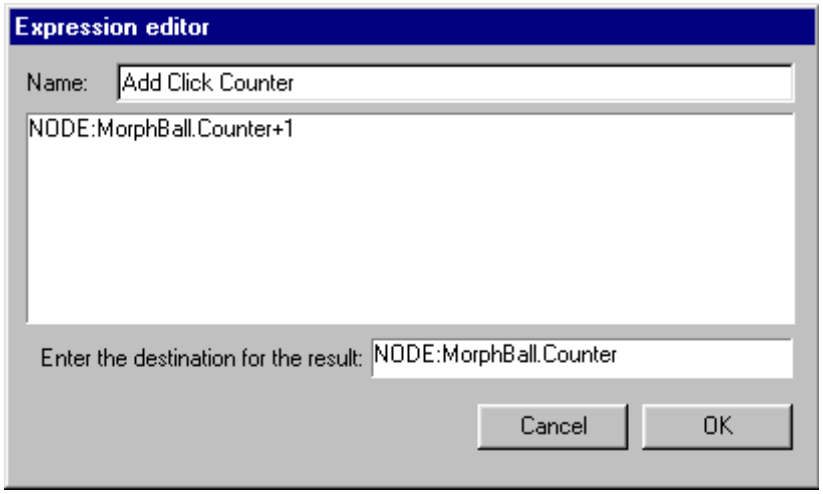


Figure 16

To use an actual Expression, you just drag the expression from the expression window onto the Event Map. In this case, I've selected the "Add Click Counter" to drag and then drop it somewhere on the Event Map. You can see a quick example in *figure 17*.
What it does is this: If the user is clicking the left mouse button on the "MorphBall" object, then it starts the "execute expression" action, (the "AB=" icon) which executes the "Add Click Counter" expression.

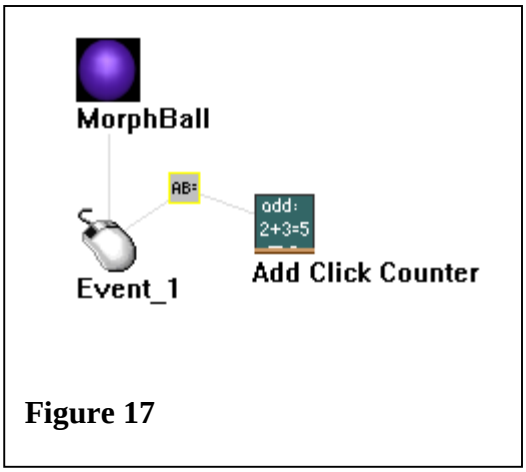
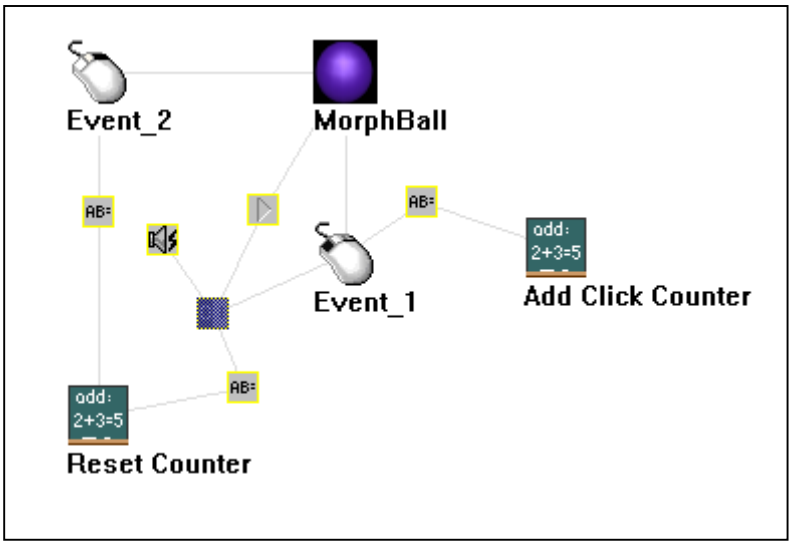


Figure 17

To explain a little better, I've made a little example of a "useful" way to use the object properties and expressions.

What it does is that it doesn't play the animation until the user has pressed the "MorphBall" object 3 times before. When the animation begins, it also resets the counter to 0…

To make this work, we need to check the object property value of the Counter, and we use the "Conditional test" action for that. (shown as a (01) icon).



Preview
Run
Start the preview in the Preview window, so one can try out all made events and actions

Stop
Stops the preview

SAVING

The saving dialog seen in *figure18* has some user-defined options on how to compress the animation data.

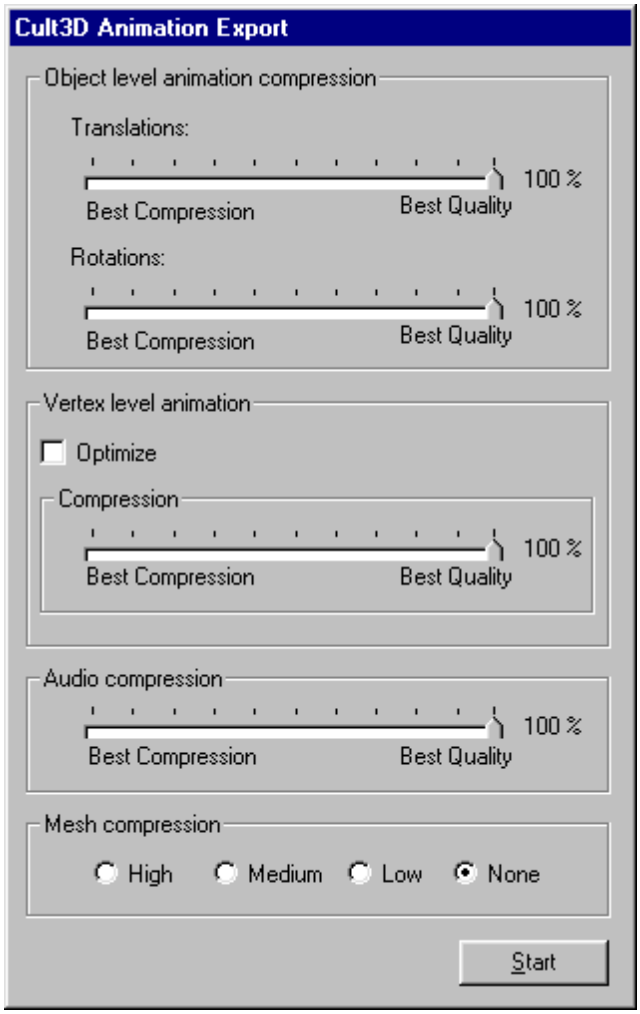


Figure 18

All the sliders in the dialog box works in the same way: values ranges from 100% to 1%, where 100% mean "best quality-zero compression" and 1% mean "low quality-best compression".

Object level animation compression.
This group is charged of controlling the compression levels for the animation matrices. "Object level" means that the animation that we are talking about is the matrix channel.
The matrix channel provides uniform rigid translations and orientations to the whole object, for each object in the scene.
Therefore, the first slider controls the compression level for the translation channel, while the second one controls the compression level for the orientation channel.

Vertex level animation compression.
Vertex level animation is a secondary animation channel: it controls motion for each vertex in each object. At this level, each vertex is associated with its own motion path. Therefore, objects containing vertex level animations are usually much bigger than the ones containing only plain object level animations.
This group contains a check box called "Optimize". If the optimize option is set, the Designer will try to export the minimum vertex level animation information required to run the current presentation.

The "Optimize" option is particularly efficient when the animations in the presentation are controlled by "Animation ”jump to”" actions.
NOTE: This option should not be selected when the slider below it is set to some amount different than 100%.

Audio compression.
The slider controls the compression amount for the digital sound (WAVE files). MIDI files won't be affected by this setting. The frequency doesn’t get changed and neither do we change a stereo sample to mono by this compression. So if the sample is 44.1kHz, stereo, then it will still be a 44.1kHz, stereo sample, regardless of the compression level. The samples will only get more and more distorted the lesser value you set the compression slider, but on the other hand, it will take less space.

Mesh compression.
The mesh compression contains a set of combo boxes. Those combos are used to set the compression level for the meshes contained in the presentation. It’s a geometric compression and does not reduce any polygons in the mesh, only the geometric precision of the objects. The compression algorithm isn’t lossy in any other way than it can lose precision of the objects geometry data.
Losing precision of the geometry can generate small visual artifacts on some objects using a high compression. So if you notice some visual artifact on your object, then use a lower mesh compression

CULT3D JAVA API

This documentation can be found were you installed the Cult3D Designer program. It should also be found on the start menu were you placed the Cult3D Designer program.

THE MINIDISC PLAYER

The scene graph includes all the scene data included in the Cult3D file, such as cameras and object information. If the objects are placed in a hierarchy, the hierarchy is shown in the scene graph. An example of a minidisc object, *figure 2* shown in the simulation window has a scene graph shown in *figure 1*.

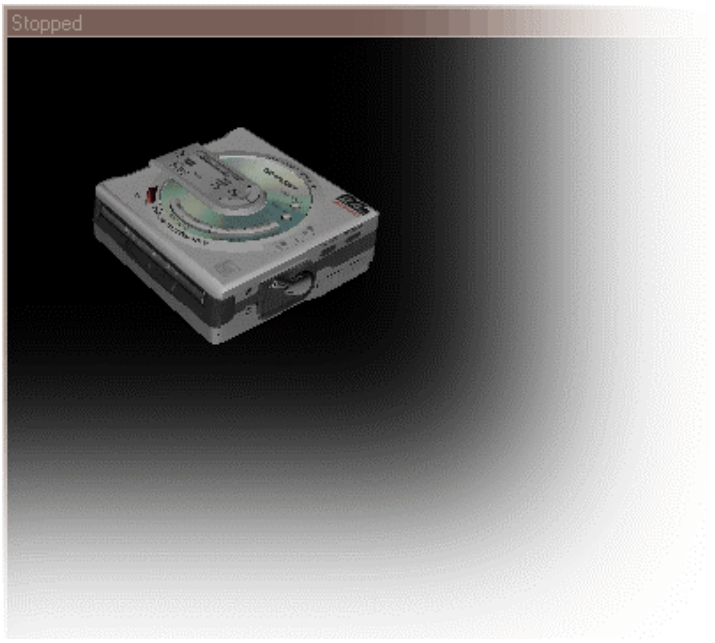


Figure 1 - Minidisc object

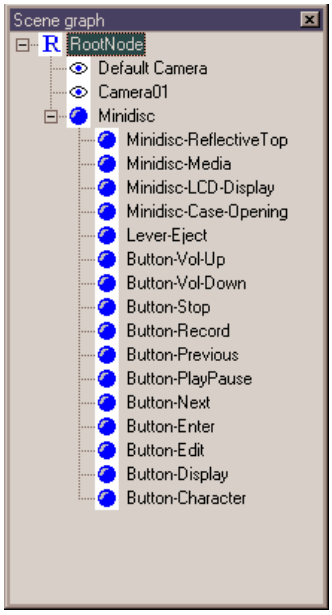


Figure 2 - Minidisc

To define the events to specific objects, one drags the object name shown in the scene graph or one can click with the mouse button on the selected sub-object in the simulation window while holding down the ctrl-key and dragging the content to the Event Map and release. Then an icon is created with that selected sub-object. When the specific sub-object is shown in the Event Map, one can drag that object onto a created event to further define the scene.

Figure 3 - Event Map

In the Event Map a connection is shown by a drawn line between the object and the event connected to it. So, if for instance we want to make the “Button-PlayPause” accessible with a left mouse button event, the

Event Map can look something like in *figure 3*.

With the action menu shown in *figure 4* one can define actions to the events created for the specified objects. For this example we want to make a "play sound" action.

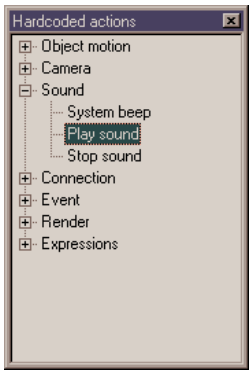


Figure 4 - Actions

Just drag the "play sound" text to the Event Map area and release it onto the event we just created ("Event_1") which is assigned to the "Button-PlayPause" object.

To load a sound resource into the scene, we need to go via the "View" menu and select Sounds. From that dialog we can add all the sounds we want to include in this scene.
When the sound is loaded (either a sample or a midi song), just drag that line onto the Event Map. There a sound icon will be created with the name of the sound added. When that is done, connect the sound to the "play sound" action.

When all this is done, the Event Map should look something like in *figure 5*.

But of course we need to define the "Button-Stop", so that the playing music stops.
To do that, just drag the "Button-Stop" object from the simulation view while holding down the ctrl-key (if you just press with the left mouse button on an object in the simulation view, you'll easier see what you've hit in the Scene Graph) to the Event Map area and release the buttons.
Add another Left Mouse button event.
Connect the "Button-Stop" to the newly created event. ("Event_2"). Add a "stop sound" action to the event. Drag the sound you want to stop playing to the stop sound icon. In this case, "canyon".

Figure 5 - Event Map 2

Remember that all the sounds one uses are included into the Cult3D file, so if you define a lot of samples in for example full CD quality (16bit 44khz, stereo), then the file isn't going to be very small.

Included into the Minidisc scene, we have the actual media. What we would like to do here is to eject that media (disc). When the user press the eject lever, so lets start of with selecting the "Lever-Eject" object and drag that to the Event Map area while holding down the Ctrl-key.
Add one more Left Mouse button event. Connect the "Lever-Eject" object to the new event.
The particular actions we're going to use are "Translate" which are found under "Object motion" as seen in *figure 6.*
Drag the Translate action to the Event map and release it onto the new event icon.

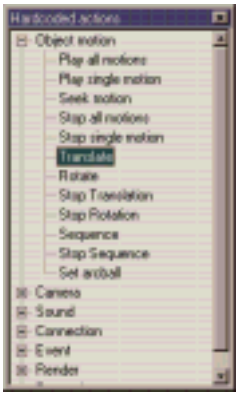


Figure 6

Since we can't see the "Minidisc-media", we have to select that object from the scene graph instead of dragging it from the simulation view window. Just select and move, and release it onto the Event Map area.
Then drag that object and release it on the Translate action to make the connection between those two.

Double-click on the Translate action to access the options to define the translation values. Remember that these values use the same unit-definition in your 3d-modeling program. One unit there, is one unit in Cult3D Designer, see *figure 7.*

In this case, we will move the "Minidisc-media" –2 units in the Y axis. We would also like to "play" this translation during two seconds, by writing 2000 in the "performance-duration" edit box. Since it is specified in milliseconds, we write 2000 instead of 2, when we want two seconds.

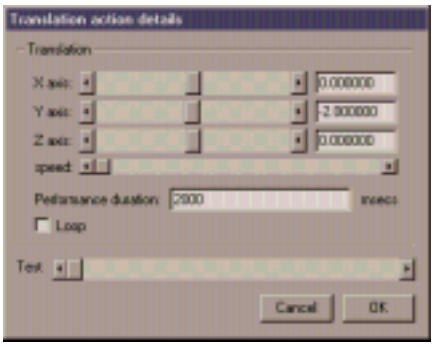


Figure 7

Then if we want the use to be able to re-insert the "minidisc-media", we add that function if the user presses the actual "minidisc-media" object.
To do that, we add another left mouse button event to the Event Map area. Then we make a connection by dragging the "minidisc-media" icon to the new event. Then we add another translate action to that new event, and via the option menu, we write 2 units in the Y axis and set the "performance-duration" time to 2000msec. This is a relative animation, so what you'll see in the simulation view is that the "minidisc-media" object is on the other side of the minidisc. But that is correct, because during a relative animation, we have defined the media to move 2 units, which brings the media forth from the inside of the minidisc, and to then re-insert it, we move it 2 units the other way.

The result should look something like *figure 8.*

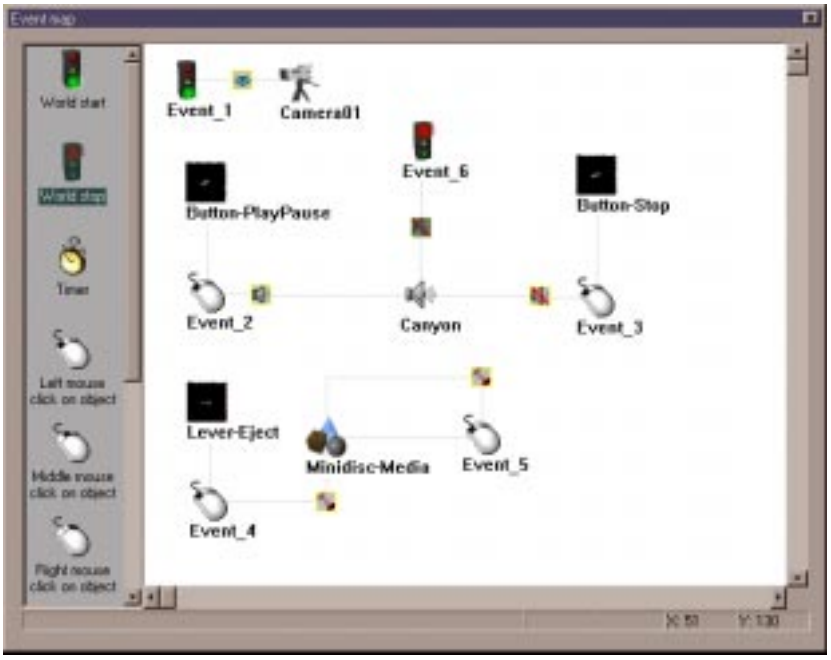


Figure 8 -

There are two events here that we haven't shown how or what they do, and those are Event_1 and Event_6, which represents the "World start" and "World stop" event. With the "World Start" event, we selected the camera we're going to use and the "World Stop" stops the sound when we quit the simulation.

THE MORPH BALL

First thing to do is load the ballmorph.c3d file into the Designer. Just go to File -> Load. Find the file and press open.

Now your work area should look should be completely empty, except you'll see a ball in the preview window.
If you click on the root node in the Scene graph, it should look something like *figure 1*.

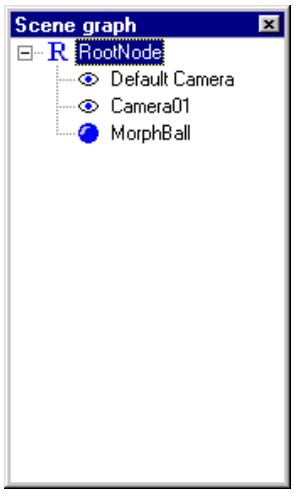


Figure 1

As you can see, we have a camera called "Camera01"and an object called "MorphBall".

We will begin to add the World Start Event to the Event Map, and from there add the select camera action to that event.
Then we will drag and drop the camera called "Camera01" out onto the Event Map and drop it onto the select camera action to make the connection.
Now we have defined to use that camera in this presentation.

We would also like the user be able to rotate the object, and to do this, we bind an arcball to the "MorphBall" object. Find the Bind Arcball action under the object motion label in the Action menu. Then drag that action and release it on the world start event we created earlier.
Then drag the "MorphBall" object from the scenegraph and drop it in the Event map area over the Bind Arcball action. Now we have defined that the user can rotate the "morphball" object.

The object "MorphBall" includes animation, which has 300 frames. Which shows the ball morphing first to a box, then to a cylinder and then back again to a ball. What we would like to do is split this whole animation into 3 sub-animations, so we easier can control them. To do this, right-click on the "MorphBall" object and click on the "Edit motion keyframes" option.
Here is the Motion Editor, which lets you define the different motions.
We will do 3 sub-motions, which are 100 frames each, as one morph takes 100 frames.
First of we create the Sphere to a Box sub-motion by clicking on Create, then we write the name of that first sub-motion. Let's call it "1. Sphere-Box", then we define the start keyframe to 0, and the End keyframe to 99.

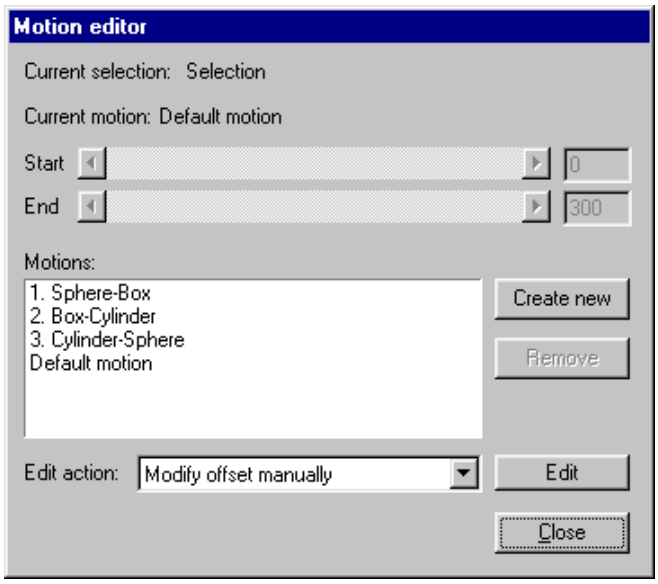


Figure 2

Then we do the other to, but call them “2. Box-Cylinder” which goes between 100-199 frames, and then the “3. Cylinder-Sphere” which goes between 200-299.
The Motion Editor should look something like *figure 2.*

Then we want to play these motions when the user clicks on the object with the right mouse button, but we only want to play one sub-motion on each click. So what we do is add 3 new “Right mouse button click” events to the Event map, and connect a play motion to each event, and connect the “MorphBall” object to each play motion action. Then your Event Map should look something like *figure 3.*

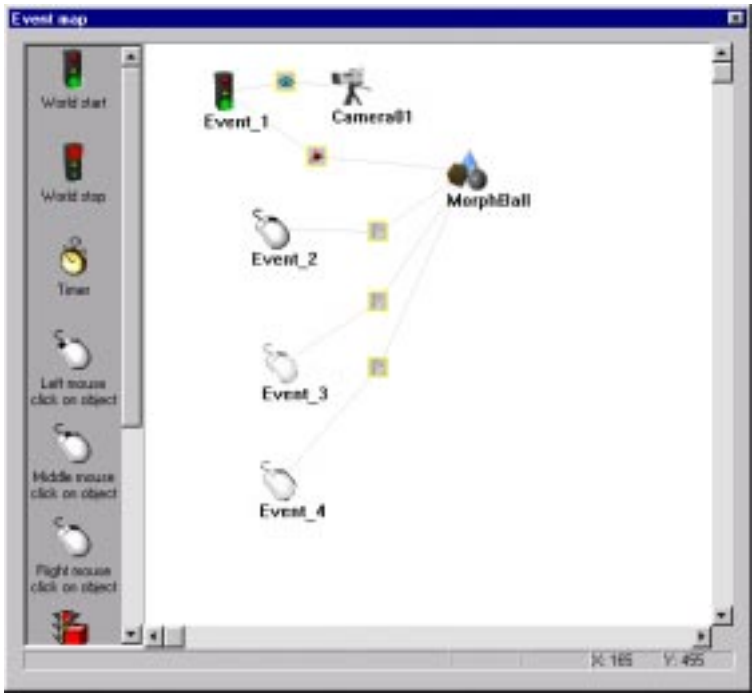


Figure 3

As you probably have seen, 2 of the new events are “shaded”. This is because those two have their “initial activation” option unchecked. To do, this, right click on the event, and click on the already checked initial activation option to uncheck it. This is done, because we don’t want to activate the others before we have played the first one.
And when we have played the first one, we want to deactivate that one, so it doesn’t play its motion until the others are done.

To do this, we first find the Deactivate and Activate Event actions in the Action menu. You can easier see in *figure 4.* Then you connect those as you can see in *figure 5.*

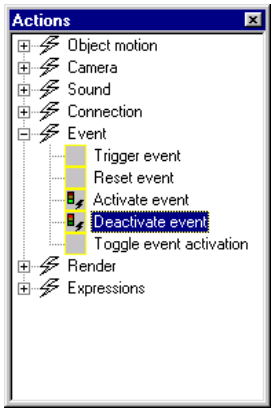


Figure 4

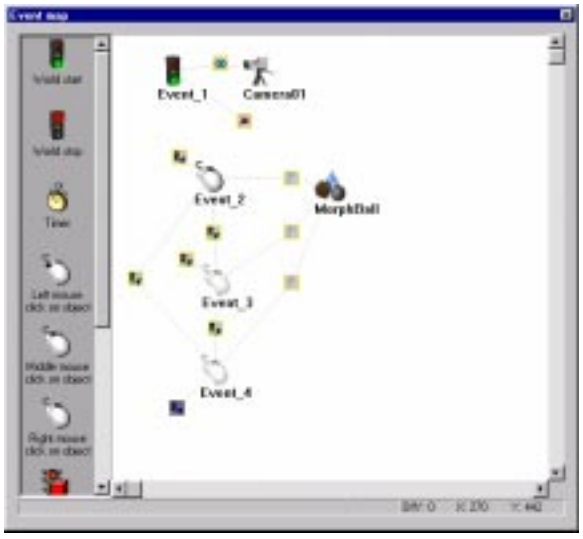


Figure 5

Remember though that the deactivate action should deactivate the event it’s attached to, so just drag that event onto the corresponding deactivate action. You will probably see a double line going to and from that event and action.

APPENDIX A—CULT3D DESIGNER FILE FORMAT

Cult3D's fileformat is a proprietary format of Cycore Computers. It's a "closed" standard for other people & companies to read, manipulate or change. The reason for this is to be able to publish ones work (3D objects and animation) without being scared that the files data can be manipulated and used in other circumstances. In other words, a *.CO file can naturally be scattered around the world if published, but inside the file are all the information safe. The *.Co file includes desired information like author and organization copyrights. (Which are written into the file when creating the Cult3D file.)

Cult3D Designer has six different types of files that it uses. These are:

*.C3D – Editable Cult3D Object file
These files are what you get from a Cult3D Animation Exporter from a 3D modeling/animation software if you're interested in getting both object and animation data and which to edit in "Cult3D Animation Authoring Tool".

*.CO – Distributable Cult3D Object
These files are the Cult3D objects, which you can "safely" distribute. These files can't be loaded into the "Cult3D Animation Authoring Tool". It can only be viewed in a Cult3D viewer. Such as a Netscape or a Internet Explorer plugin.

Both *.c3d and *.co files can include meshes, textures, materials, sound, Java classes, animation and interactivity data.
Basically they are the same, except the *.co files are "safe" to distribute.

*.C3P – Cult3D Project File
These files are used to save and load the settings and work of your interactivity you've defined in the "Cult3D Animation Authoring Tool". Placements of the icons, connections etc.

*.WAV – Sound files.
"Cult3D Animation Authoring Tool" can handle .wav files to include sound.
Note: Even though we compress the sound files, and get an average of about half the original filesize. You should use long and "cd-quality" (44.1kHz, 16bit stereo) sounds with caution, because they can add a lot to the filesize of your presentation. One second of cd-quality samples takes about 170 KB raw, and compressed in the Cult3D files, 80-90 KB.

*.MID – Midi files

*.CLASS – Compiled Java Class.
To add a customizable action, you need to compile Java classes to do it…

APPENDIX B—CULT3D PLAYER SPECIFICATIONS

Rendering modes

- Wireframe
- Constant shading
- Flat shading
- Gouraud shading with phong highlight
- Phong shading

Texturing

- Arbitrary sized textures; memory is the limit. (power of 2 sizes restriction)
- Perspective correct texture mapping
- Sub-texel (texture map pixels) accuracy
- Point sampling
- Bilinear interpolation

Rendering methods

- Single color
- Bump mapping
- Environment mapping
- Transparency
- Sub-pixel correction (accuracy)
- Z-buffered background image

Rendering features

- 24-bit, True Color rendering
- Auto-converts to the color depth the OS is currently using, with additional dithering in 16 and 8 bit modes.
- Optimized blitting for maximum performance
- Unlimited number of polygons in an object
- 32-bit Z-buffer
- Cross platform support with hand tuned assembly optimizations
- Hierarchical worlds and objects for easy and fast animation
- Optimized streamable format structure for Internet use
- Stereoscopic rendering, for use with Kasan 3D-MAX LCD shutter glasses.

Supported platforms

Netscape

- Windows 95/98
- Windows NT
- MacOS (PPC)
- HP-UX 10.10 or later
- Solaris 2.4 (SPARC) or later
- AIX 4 or later
- Linux 2.0 or later

Internet Explorer

- Windows 95/98
- Windows NT
- DEC Alpha Windows NT

Opera

- BeOS

APPENDIX –C—CULT3D DESIGNER SPECIFICATIONS

Interactive Development Environment

- Event driven architecture.
- Simulations are easily built using drag and drop visual tools.
- Objects properties can be modified during simulation with expression type actions.
- Real-time feedback.
- Hierarchical key frame based animation are supported.
- Capable of importing all animation sequences from programs such as 3D Studio Max.
- Sub-segments of motion can be derived from the original animations.
- Timed morph-based transformations between key frames are provided.
- Possibility to add timed translations and rotations.
- Animation are kept synchronized to the given frame rate.
- Animation's time of execution can be re-targeted to different timings.

Simulation

- Event-Action interface.
- Base events are provided (i.e. timer, mouse click, …).
- Java support. Java coded actions can be created to provide the simulation with the highest level of procedural support.
- Timed action-sequences.

Other

- Import hierarchical models, cameras and animations from programs such as 3D Studio Max.
- Support progressive download of object, texture and motion data.
- Possibility to decide which simulation part must be downloaded first.
- Java-class callbacks supported.
- Support for anchor actions to connect to a web page or a new simulation file.
- Advanced polygon reduction.
- Advanced polygon compression.