WINGED EDGE TECHNOLOGIES

MIRAI 1.1
WHITE PAPER

This document introduces you to some of the revolutionary workflow concepts found in Mirai, Winged Edge Technologies' professional 3D content creation tool.

Image by Bay Raitt

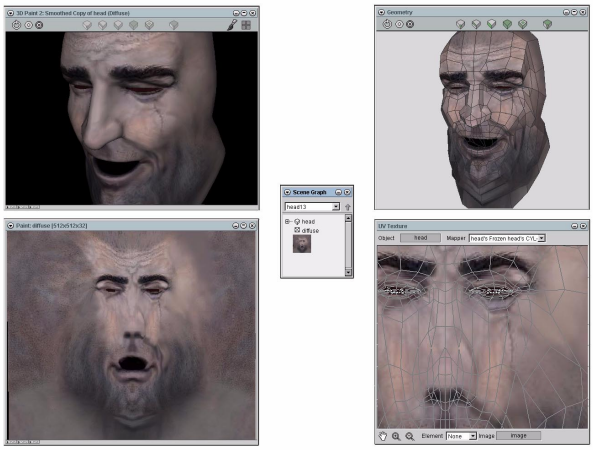
INTENDED AUDIENCE

This document is intended for readers who want to learn more about how Mirai differs from other animation systems. It explains some of the concepts behind Mirai and gives an overview of the various editors found within the package.

A TRUE MULTI-EDITOR WORKSPACE

Mirai was built from its inception to be a *complete environment* dedicated to 3D authoring. This is significantly different from the traditional "single document" editing approach currently found in most software programs.

In the Mirai environment, you can have any combination of editors open simultaneously, including multiple copies of the same type of editor, editing the same data from distinct viewpoints. Consider a typical Mirai desktop:



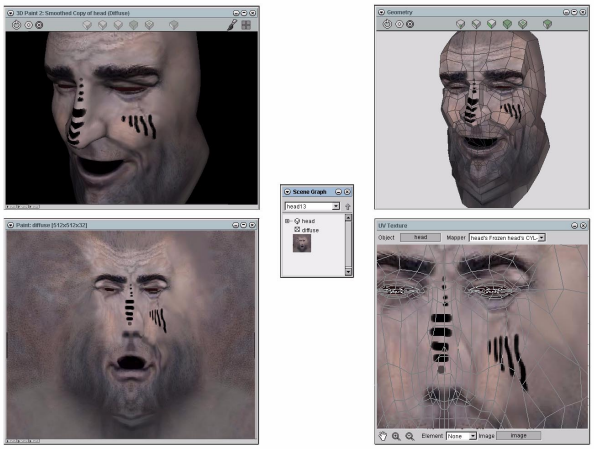
This artist has opened different editors to 3D Paint, model, paint, and edit the UVs of a character simultaneously. In Mirai, you're able to edit multiple aspects of the same entity at the same time.

MIRAI'S MULTI-THREADED ENVIRONMENT

Because Mirai was designed from the beginning to be *truly* multi-threaded, any changes that you make in one editor are immediately reflected in any other editor using a "scene-locking" mechanism.

In Mirai, a workspace configuration does not represent a "mode," or distinct functional area of the software, but rather a layout of separate editors all touching the same elements in the scene.

Each editor is a tool with a specific purpose. The tool updates the entities directly and in real time; change an entity in one editor, and any other editor touching the entity updates itself automatically. The result is that your editors always remain synchronized.



The result of such a model is complete flexibility to tailor your environment to fit the task at hand. Quite simply, you can open any combination of any editors at any time. In those editors, he can edit any aspect of any element, and can even edit the same element simultaneously in multiple editors of the same or different types. For example, you could edit the facial animation of a character in one Animation editor while viewing the scene motion in another. Changes that you made to the facial animation would appear instantly in the animation viewer containing the scene animation.

MANAGING COMPLEX PRODUCTIONS: SCENES & SUBSCENES

In Mirai, the basic unit of data is a *scene*. When you start Mirai, an empty, unnamed scene is created for you. Any elements that you create when a scene is loaded are *added to the scene*.

When you save a scene, any modifications you've made to elements in the scene are saved.

SUBSCENES

To help better manage your production pipeline, Mirai supports the concept of *subscenes*.

The scene/subscene structure means that entire productions can be built in smaller components, then linked together in one “master” scene.

In its practical application, subscenes could be used like this:

- Animator #1 creates the master scene for the overall animation (“main”).
- Animator #2 opens a second scene, creating a character’s skeleton and skin (“character”).
- Animator #1 then loads “character” as a subscene into “main” and begins to keyframe animation for the skeleton.
- Animator #2 continues to modify the skin and muscle. When Animators #1 reloads his scene (which reference the subscene), the changes that have been made to the character are propogated into “main.”

This unique scene and subscene hierarchy is made possible through the management of entity *links* which are maintained both within and between scenes and a concept called *shadowing*.

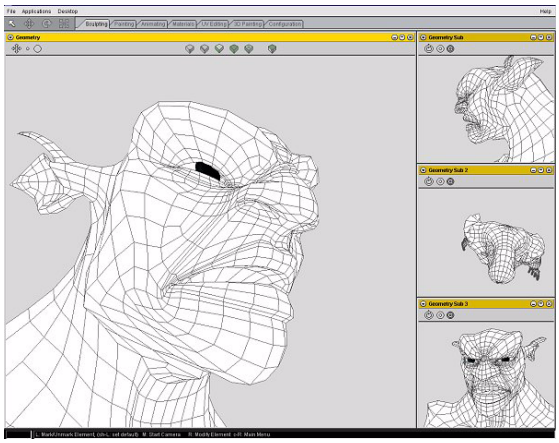
Scene entities use these links to record their relationships to each other. The links in a subscene, however, can be shadowed by links in a parent scene. The result is that an animator can load a subscene, and make assignments to elements that overshadow (but don’t replace) the links in place in the subscene.

For example, if you assign a red surface material to a ball in your scene, someone in a parent scene can re-assign a yellow surface material to the ball. The ball is only yellow when the original scene is loaded as a subscene. If the original animator loads the scene, his original links are maintained. The same type of linking can be used for animation of scene elements.

MIRAI’S HUMAN INTERFACE

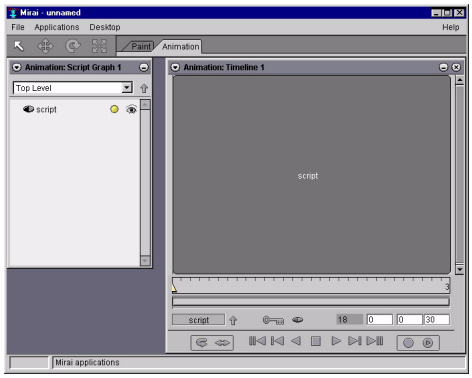
Mirai uses a familiar desktop and window metaphor for visually organizing the workspace. Top level menus appear on the main menu bar, while editor specific menus appear within each editor. Mirai’s desktop and human interface elements are platform independent; they’ll have the same look and feel regardless of the platform you’re on.

Mirai lets you define selectable configurations which are maintained from session to session. The user defines which editors are opened when the configuration is selected. Additional editors can be opened at any time.

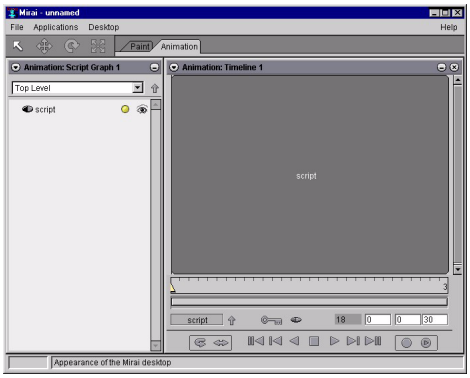


One nice feature of Mirai’s desktop is its ability to resize selected windows either horizontally or vertically so that they “bump up” against other windows. This makes it easy to efficiently use all of your desktop. We call this “fitting” the window.

Consider the configurations in the following images. There’s wasted space in between the Script Graph and the Timeline window which could be used by one of the two editors.



A single click on the title bar expands the window both horizontally and vertically:

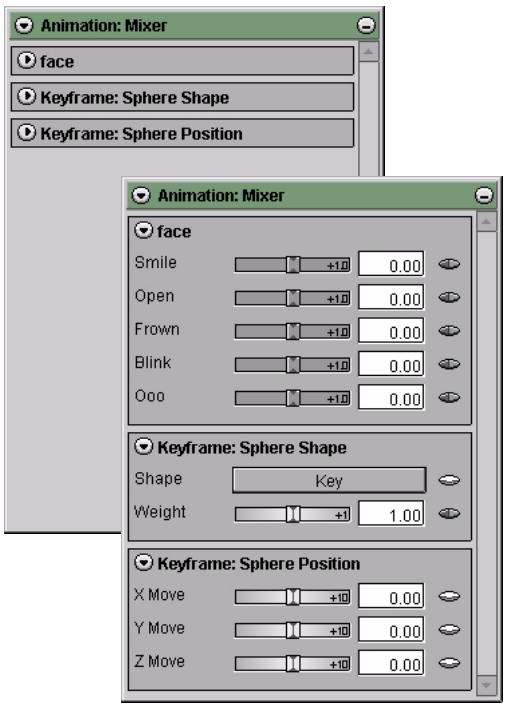


This is just an example of some of the careful consideration that went into developing Mirai's human interface.

Since you can have multiple editors open and active simultaneously, you need to tell Mirai which editor you want to work with. You do this by *moving* the mouse over the window you want to work with (no need to click on the window to make it active).

Because desktop space is at a premium, Mirai incorporates collapsing columns for several editors, including

the Animation Mixer, Paint Tool Parameters, Skeleton Properties, and Particle Group Properties menus:

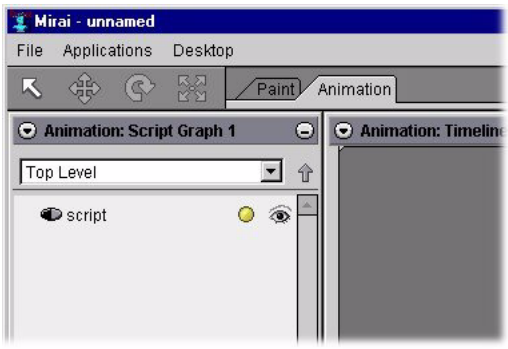


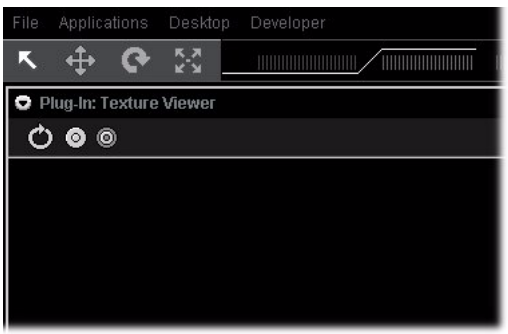
Hiding channels when they're not necessary makes it easy to maximize use of valuable desktop real estate.

Wherever appropriate, Mirai uses a custom type of dual-mode slider which we call the *super slider*. Animators can define the behavior of the slider:

- Making it a "bounded" slider sets minimum and maximum values
- Making it a "boundless" turns the slider into a shuttle jog. The further you move the handle from center, the faster the value increments or decrements.
- Where appropriate, animators can set the range and increment values for the slider, too.

The *appearance* of the Mirai workspace is highly customizable. A wide array of user-definable preference lets you customize both the appearance and behavior of Mirai:





ABOUT MIRAI'S EDITORS

You open Mirai's editors using the commands under the *Application* menu in the menu bar. You can select any of the following applications:

- Scene Graph
- Geometry
- Animation
- Paint
- 3D Paint
- Attributes
- Color Reduction
- UV Texture

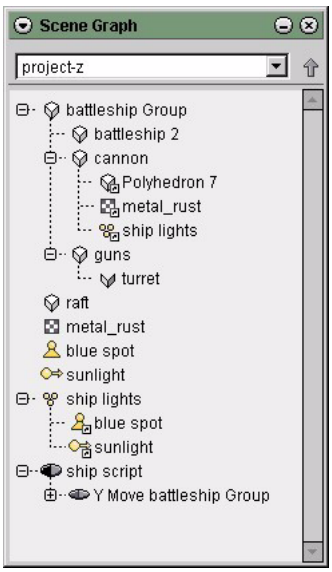
Each of these applications is made up of one or more related windows. These primary editors have other secondary editors which are applicable only to those editors. (These seconday editors are described in more detail in the *Mirai Workbook*.)

MANAGING ENTITIES: USING THE SCENE GRAPH

You can view the contents of the currently loaded scene (models, skeletons, images, scripts, materials, lights, and so forth) in a *Scene Graph*. You use it to:

- View and create object *hierarchy.*
- View and create material and light group assignments.
- Select entities to edit.

Because the Scene Graph fully supports "drag and drop" you can drag entities directly into editors when you want to work on them, or drag entities onto each other to create assignments or object hierarchy. You can have more than one Scene Graph open at a time (in case you want to examine different parts of your scene) or drag and drop elements between different parts of a scene. Each type of entity is represented by a unique icon:



MODELING: GEOMETRY

You use Geometry to build and edit *3D entities* such as polyhedra, skeletons, and wires. You add a primitive to your scene, then sculpt that shape, manipulating the object like a piece of virtual clay into the shape you want:

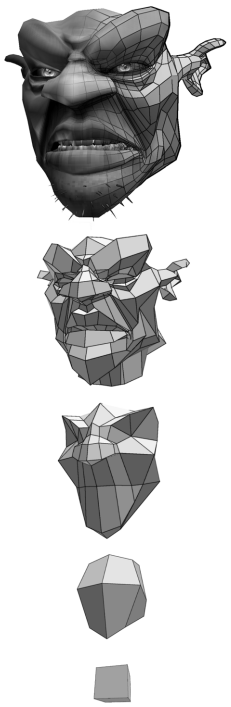
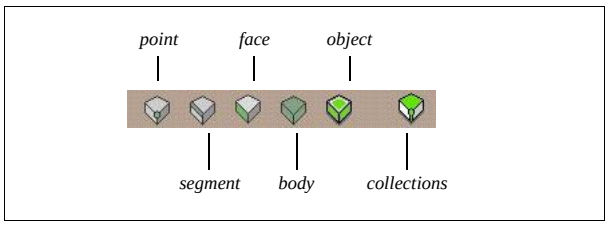


Image by Bay Raitt

You select the type of element you want to modify using the element sensitivity bar across the top of the editor. You can select points, edges, faces, bodies, or objects by choosing the corresponding icon.



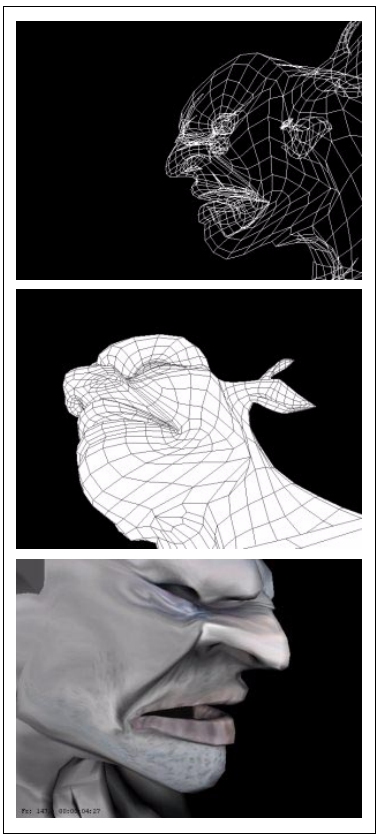
After you've selected the type of element you want to edit, you select one or more of that element type in the editor, then click once to display a dynamically created menu, containing only operations that can be performed on the selected element (or collection of elements). This means you won't have to wade through long menus with lots of "grayed out" commands.

This approach is purposely distinct from the "single-tool" approach of other systems in which users click on a tool then click on an element to modify the element with that tool.

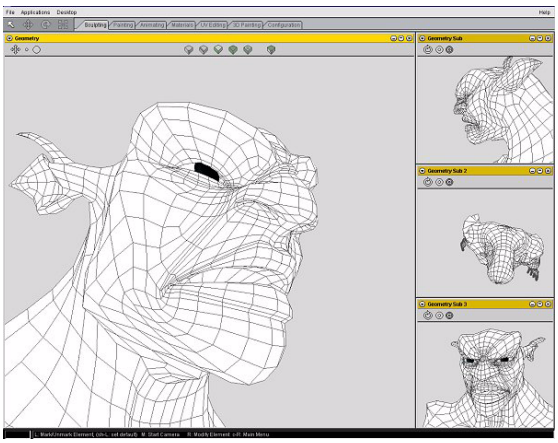
Geometry's "selection, then operation" approach lets you edit multiple elements simultaneously (for example, you can select multiple faces and extrude all the fingers of a hand with one operation). Once mastered, this method lets you speed through the modeling process.

SINGLE WINDOW PERSPECTIVE VIEW

Mirai's Geometry is based on a single window perspective view. By simulating a 3D potter's wheel, you can work more like a traditional sculptor, first pulling out a rough form, then adding detail as needed. And, like the sculptor, you constantly adjust your point of view, turning the model and resculpting the silhouette, shifting between wireframe, shaded, or textured modes with hotkey commands:



Of course, users more familiar with the traditional 4-view model (perspective plus x, y, and z views) can create a custom configuration showing those views:



GEOMETRY AND SUBDIVISION SURFACES

Part of an effective use of a polygon-based system is learning about and using its subdivision surface technology. One of the basic recommended techniques in Mirai is to sculpt, animate, and even paint a lower resolution model, a model which remains light and responsive.

Once your low-resolution model is complete, you can then generate a smooth copy, smooth the mesh one, two, three, or more times to get the desired level of detail necessary for your project:

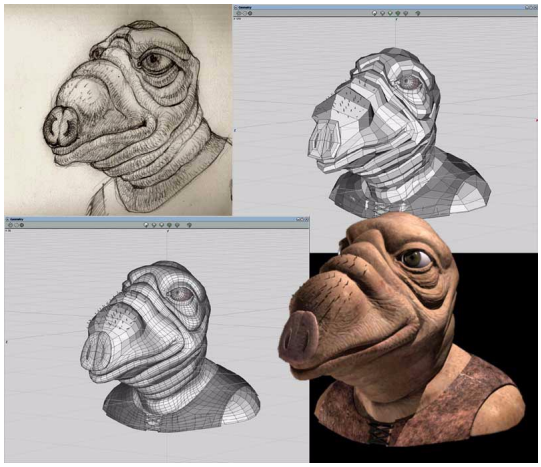


Image by Sven Jensen

By building low-polygon meshes which smooth well, users can work with lighter meshes while animating, then swap to a higher resolution model for different output targets.

Note that you can have multiple smoothed copies of an object, each smoothed a specified number of times.

CONTROL BODIES AND DISPLACEMENTS

When you smooth an object, you have the option of creating a *copy* of the original (rather than smoothing the objects itself). When you create such a copy, you also have the option of creating a *controller-controllee* relationship between the original and the smoothed mesh. When you create such a relationship, deformations made to the lower-resolution object drive the shape of the higher resolution object. For example, animating to different states (morph targets) on the low-resolution model (shown on the left) drives the higher resolution model (shown on the right):

Image by Bay Raitt

This means that you can always work with a low-polygon surface, keeping things quick and responsive, then swap in the higher resolution model when necessary for generating rendered or printed output.

POLYGONS: GREAT FOR ORGANIC AND INORGANIC MODELING

Although typically associated blocky, low-resolution models, a good polygonal modeler lets users create realistic organic and inorgranic shapes with equal ease. Most sculpting operations in Mirai's Geometry can be modified to use an area of falloff. These *magnet* modifiers let you create smooth, natural surfaces which are indistinguishable from NURBS or other non-polygon output:

Image by Sven Jensen

Smooth, machined surfaces can be generated with equal ease:

Image by Bay Raitt

ANIMATING

Mirai's approach to animating supports traditional animators while introducing innovative techniques that expand the your artist's toolchest.

Animation is recorded in a *script*. Scripts are composed of channels. Timing can always be manipulated independent of content; slide channels, keyframes, or entire scripts along the timeline, or add channels to edit the actual flow of time (for example, you can put a slow-in curve on the timing of the *entire* script). You can add any number of scripts to a scene.

The ability to mix and match skeletal keyframing and motion capture data without restriction makes Mirai an incredibly flexible solution. Motion capture data can now be thought of as a malleable commodity, easily "tweakable," rather than a rigid piece of canned animation. The ability to blend in artist-defined poses over such data makes motion capture a reusable resource which can be used again and again in any number of different projects.

Motion capture data can be imported/exported in Acclaim, BioVision, and Motion Analysis formats.

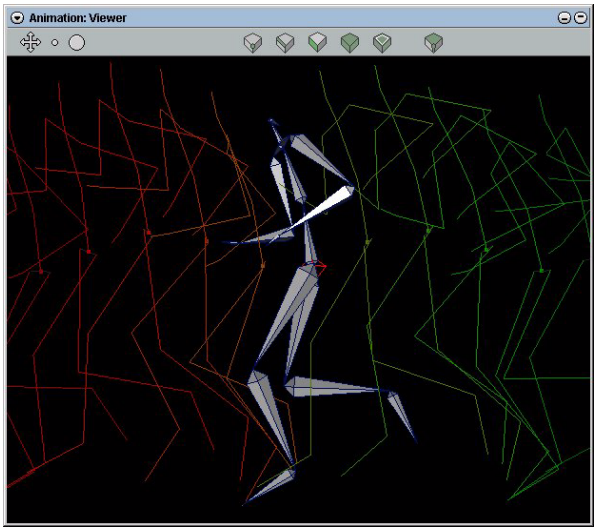
When you start the animation editor, four windows are created:

- Animation Viewer
- Timeline
- Mixer
- Script Graph

Just as you can have many scripts in a scene, you can also have many Animation Viewers open at once editing those scripts. This is particularly useful when *chaing* scripts together; you can edit the "called" script in one editor, then scrub through the main script and see those changes reflected immediately.

THE ANIMATION VIEWER

The Animation Viewer window is similar in appearance to the Geometry window. In the window below, the onion-skin feature has been turned on for a skeleton, a great tool for keyframers:



Unlike Geometry, where you can do any type of modeling, you are limited to performing only operations which can be *animated* in the Animation Viewer. This concession allows scenes to be animated at a much higher rate, since the versions of the objects being displayed are "lighter" and don't carry all the topographical information that exists for the object when it is loaded into Geometry.

You can manipulate objects in the Animation Viewer using the same "selection, then operation" method described earlier for Geometry, or you can use the animation tools just below the main menu.

The standard set of animation tools can be found in the upper left corner of the desktop, under the menu bar:

The tools have the following functions:

Table 1.1 *Animation Tools*

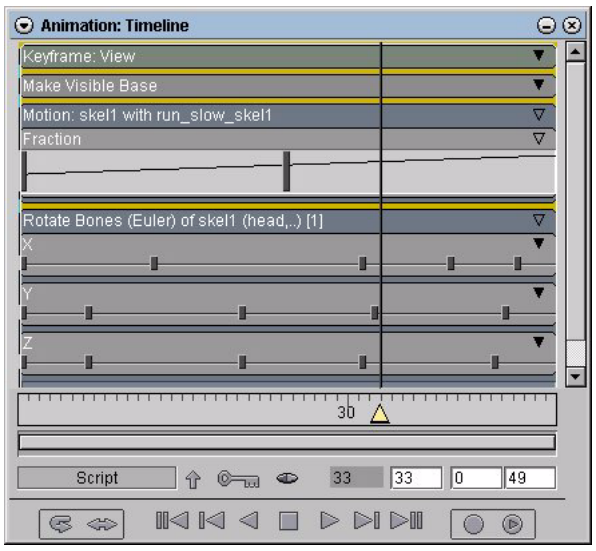
	PICKER TOOL	Used to select elements; this is the "default" tool for most of Mirai's 3D editors.
	MOVE TOOL	Used to move (translate) the selected elements.
	ROTATE TOOL	Used to rotate the selected elements.
	SCALE TOOL	Used to scale the selected elements.

These tools can be used to manipulate objects in the 3D viewer. The animation tools are on the desktop (rather than inside of individual editors) because they are used by *multiple editors*.

When you animate a script, channels are evaluated from top to bottom at each frame. Channels can also be nested; channels inside other channels are evaluated first. At the beginning of each frame the script is initialized. Each channel performs its operation on a specified entity.

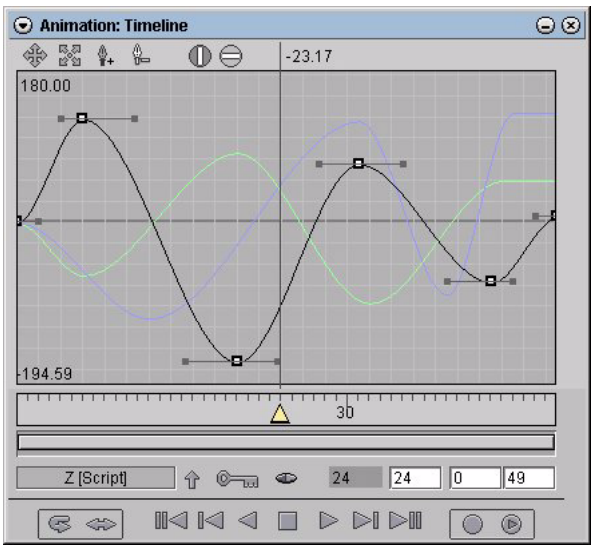
When you start Animation, a script is created for you automatically. Channels in a script appear in the channel editor portion of the Timeline.

The curve data for any operation is nested inside a channel within the operation's channel.



To appreciate Mirai's animation script system, it helps to understand that a script is simply a *list* of commands. We

call these commands *operations* (Mirai supports over 170 different animatable operations). If an operation can be animated, its curve is displayed in Mirai's curve editor:



Changes that you make in the curve editor interactively update the entities in any associated Animation Viewers.

KEYFRAMING

The primary tool for entering keyframes is located at the bottom of the Timeline:

keyframe button

The keyframe icon lets you do any of the following:

- Keyframe selected *objects*
- Insert keyframes in the selected *channels*
- Keyframe *modified* objects

Mirai also features an autokeyframing mode: go to a frame and make your changes. As soon as you advance to any other frame, those changes are recorded.

If you perform a keyframable action that can be recorded in more than one channel, it is inserted into the channel closest to the bottom of the script.

If you try to record a keyframe, but have not yet created a channel to hold its value, Mirai creates keyframe channels for you automatically.

The type of keyframe channel that is created depends on the last action performed and the type of entity you performed it on:

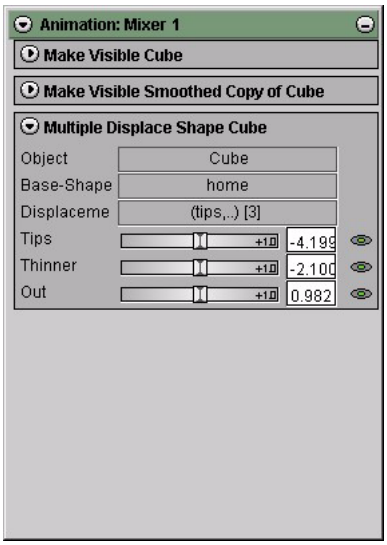
Table 1.2 *Automatic keyframe channel generation*

OPERATION	ELEMENT	GENERATES CHANNEL
Any	Vertex, edge, face, or body	Keyframe: Shape
Move	Object	Keyframe: Move
Scale	Object	Keyframe: Scale
Rotate	Object	Keyframe: Rotate
Move*	Camera	Keyframe: View
Any	Joint, bone, or skeleton	Keyframe: Pose
Modify attributes	Object	Keyframe: Attributes

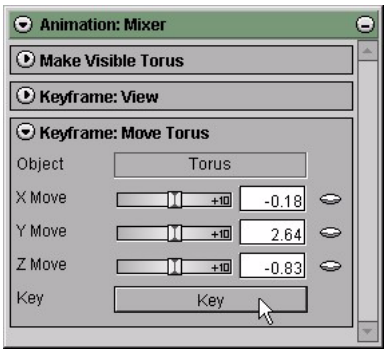
Note that if you use the animation tools instead of an operation to manipulate some element in the 3D viewer, then hit the keyframe icon, an appropriate keyframe operation is automatically added (e.g., a *Keyframe: Move* operation would be added if you had moved an object with the Move tool).

THE MIXER

You can also use the Mixer to edit your animation. The Mixer shows the dynamic values being passed to each operation at the current frame. You can also use the Mixer to insert a cue into a channel at the current frame:



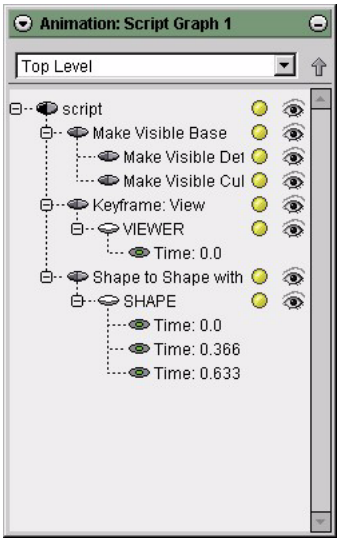
For every dynamic operation (those that take curves) one or more sliders are added to the Mixer. As you move the sliders in the channel, the animation viewer is updated interactively; when you let up, the value is recorded for that frame. You can also type in values for any dynamic operation. Finally, you can also record keyframes in the Mixer using the *Key* button found in any keyframable operation channel:



THE SCRIPT GRAPH

The Script Graph lets you visualize the *hierarchy* of a script, irrespective of time. You can also use the Script Graph to hide and show or activate and deactivate channels in a script (changes that are reflected in the Timeline and Mixer).

A Script Graph looks like this:



You can open and close portions of the hierarchy in the Script Graph in the same way you open levels in the Scene Graph.

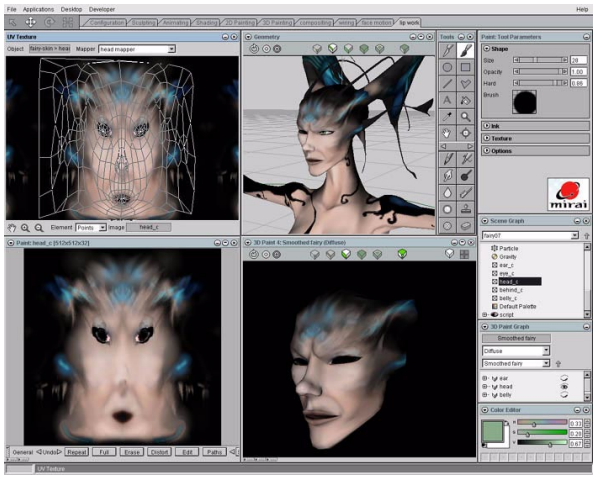
PAINTING & TEXTURING

The painting and texturing tools in Mirai let you generate incredibly realistic surfaces for your models:

Image by Sven Jensen

Mirai's non-linear system lets you paint and apply textures using a variety of techniques at any point in the character creation process. In the illustration below, for example, the artist has four editors open:

- UV Editor
- Geometry
- Paint
- 3D Paint



Changes he or she makes in one editor are reflected immediately in any other open editor. So, for example, if you used the dodge tool in the Paint application to create some highlights on a map, those highlights would show up wherever the map was being displayed (e.g., in the Geometry, UV Editor, or 3D Paint editors).

Mirai offers an extensive set of editors for painting and texturing models:

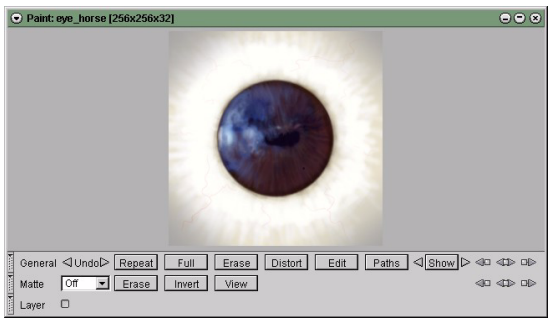
- Paint offers full 64-bit image editing capabilities
- 3D Paint: 3D Mode lets users paint directly onto the surface of 3D objects
- 3D Paint: Map Mode lets user paint on maps with UV overlays
- UV Texture Editor lets you edit UVs of applied textures
- Color Reduction lets you color reduce single or multiple images, or map multiple images into a single super palette.

In addition to the standard set of tools, several plug-ins have been developed to speed the process of texturing a model:

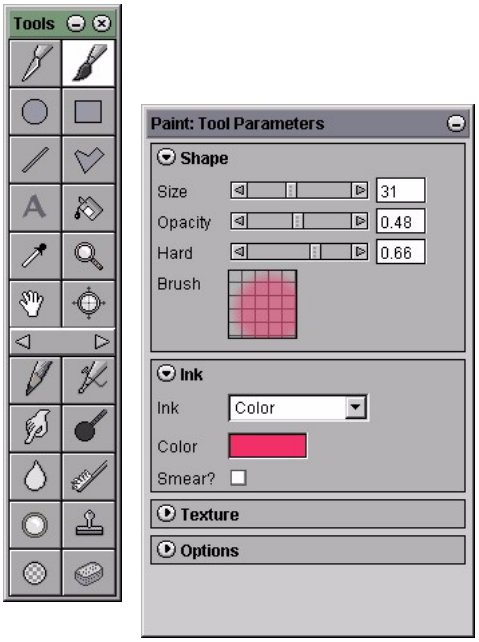
- The Snap Map Plug-In lets you “snap” an image onto all or part of a model
- The Map per Face Plug-In lets you assign different maps to individual faces (great for doing detail work with lots of small textures) with the ability to edit and rotate such textures.

PAINT

You use Paint to create and edit images. Paint displays the image you want to edit in an *image window*. The image window consists of a viewing area (where the image is displayed) and three collapsing bars (*General*, *Matte*, and *Layers*) which let you perform canvas and general image management tasks:



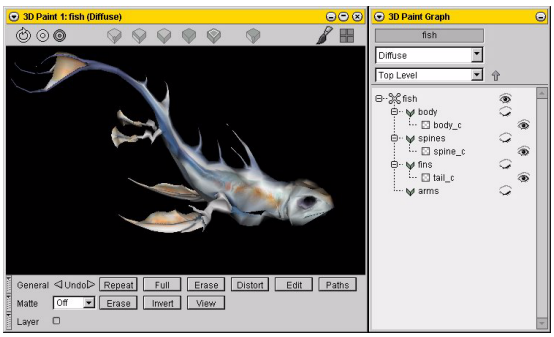
You use the Paint Tool Parameters and Paint Toolbox windows to select pre-existing tools or to design your own:



The Paint Tool Parameters menu is included in a “collapsing view” dialog, which means that you can open or close individual sections as you like when setting parameters for the tool while conserving precious desktop space.

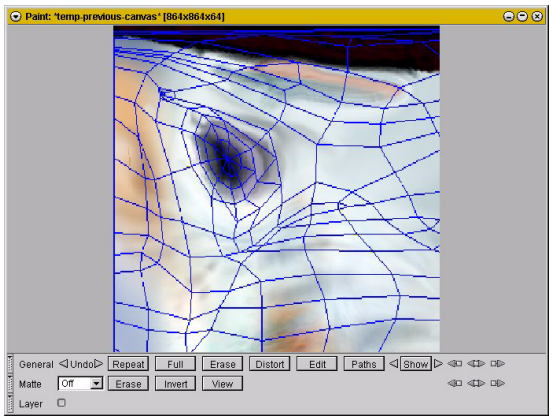
3D PAINT

You can use Mirai’s 3D paint application to paint directly on the surfaces of your models:



All of the tools from Paint are available when you’re in 3D Paint. For detail work, you can unfold the geometry of your

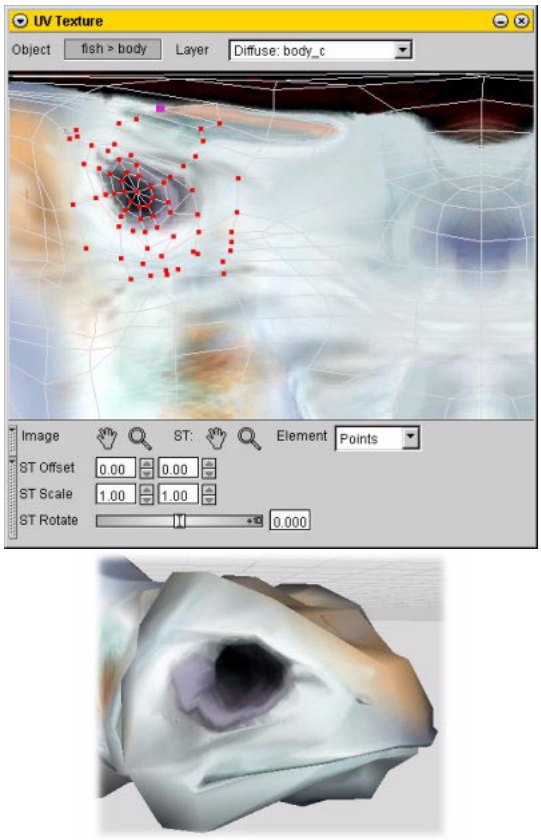
model and paint on the maps instead, displaying the UVs for all or part of the model as a guide:



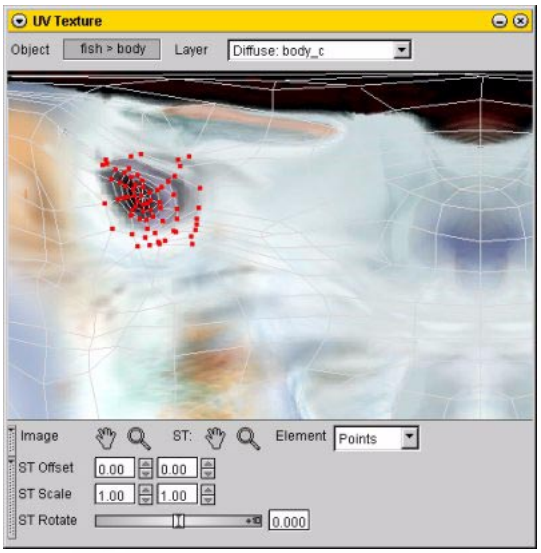
You use the 3D Paint Graph to select which object, layer type (e.g., diffuse, specular, bump, etc.), and layer you want to paint on.

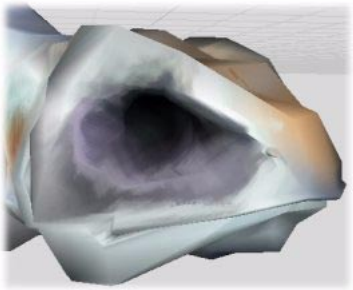
UV TEXTURE EDITOR

The UV Texture Editor lets you modify the coordinates of the textures that you have applied to your model:



You can edit UVs by moving vertices, edges, or faces Of course, since Mirai is a true 3D environment, changes that you make in the UV Editor are reflected immediately in any editor where the object is displayed:



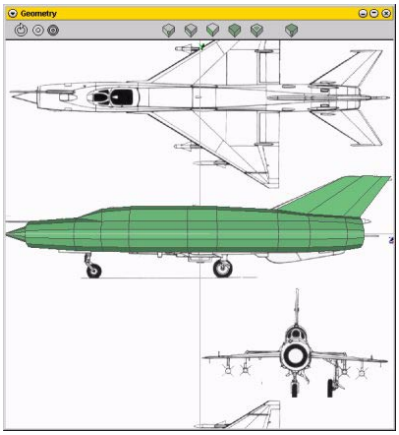


COLOR REDUCTION

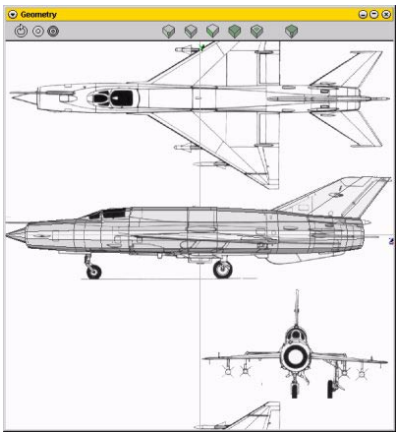
The Color Reduction application lets you *reduce* the bit depth for single or multiple images, *generate* a single, sharable palette from multiple images, or *apply* a palette to one or more images.

SNAP MAP PLUG-IN

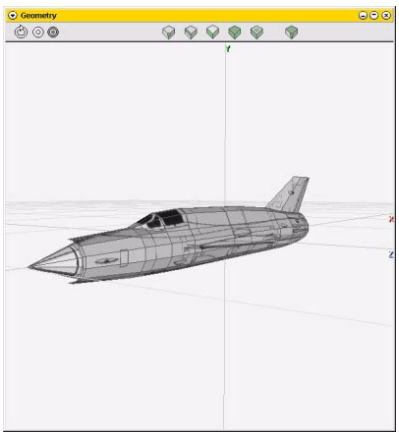
The Snap Map plug-in lets you "snap" textures displayed in a 3D viewer to all or part of the surface of a model. For example, you might first load a schematic of the model into the background of Geometry as a guide:



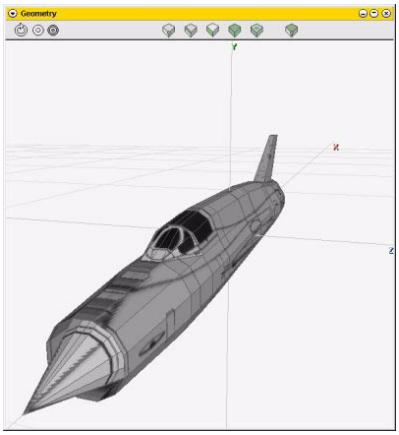
After you've lined up your model with the background texture, you can select which faces you want to "snap" the background image to (creating a new material in the process):



You can then remove the background material and cut the model in half so that only the half upon which the textures were originally projected are visible (if necessary):



The resulting object can then be mirrored around the cut face, mirroring the UVs in the process:

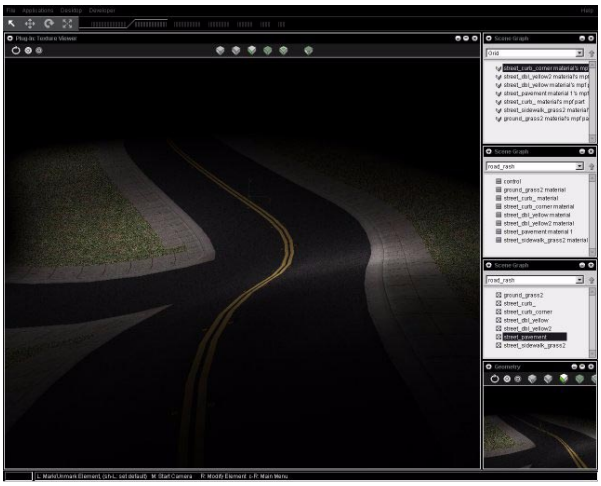


This technique can be used with both organic and inorganic models, on the entire object or selected faces. You can create any number of mappings to apply different textures to different faces in a completely interactive fashion.

MAP PER FACE PLUG-IN

The Map Per Face plug-in is intended for applications where you have lots of small maps that need to be applied to a surface.

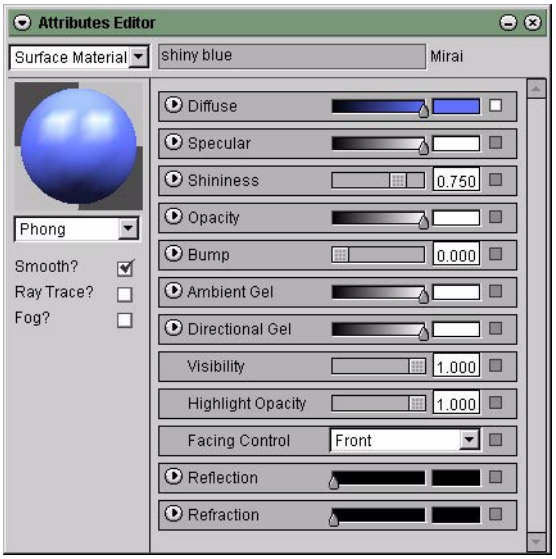
With this plug-in loaded, you can select a face and a map, and a material is created for you automatically. That material can then be applied to any other number of faces. Changing the material affects all faces to which it has been applied.



The Map per Face plug-in also adds a number of interactive UV editing commands. For example, you can rotate or flip the map projected onto a face with a single keystroke; texturing a complex surface is a breeze with this extended functionality.

ATTRIBUTES & MATERIALS

In Mirai, surface render attributes can be assigned either locally to an object or part, or created as a material and applied to any object or part. You use an Attributes editor to modify either local attributes or materials. An Attributes editor looks like this:



Mirai supports multiple render domains; this means that you can define surface attributes for multiple output targets. Translating attributes between domains is a simple command that can be executed from the Scene Graph. Supported render domains include:

- Mirai
- OpenGL
- PSX
- N64
- Dreamcast

RENDERING

Mirai's renderer supports an impressive set of features. With optional raytracing options for both surfaces and lights, a wide variety of distinctive "looks" can be achieved.

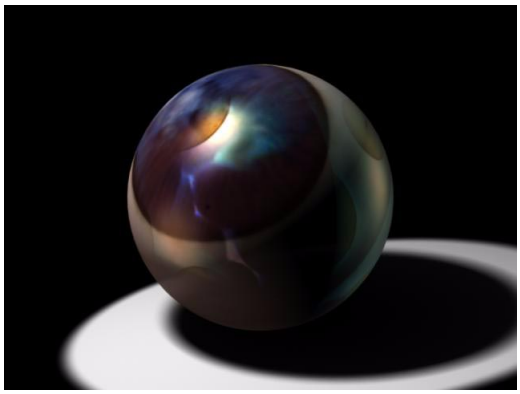


Image by Roberta Valent

Additionally, most attributes can be layered, which means that an object's appearance can be modified by blending multiple maps for selected attributes.

With great control over shadows, and additional features such as volumetric effects, oblique effects, environment mapping, fog, and depth of field, Mirai delivers professional caliber output.

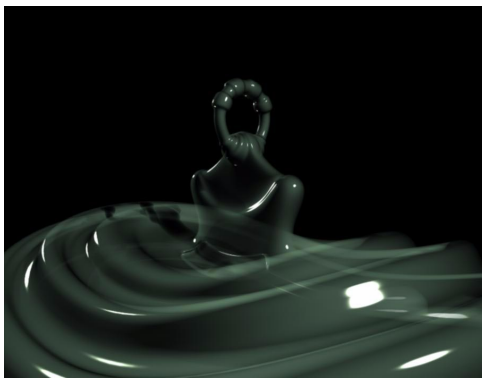


Image by Bay Raitt

LIGHTING

Mirai supports the types of lights you'd expect to find in a professional package (ambient, point, spot, and infinite). With great control over shadows and light attenuation, dramatic lighting effects are a snap:

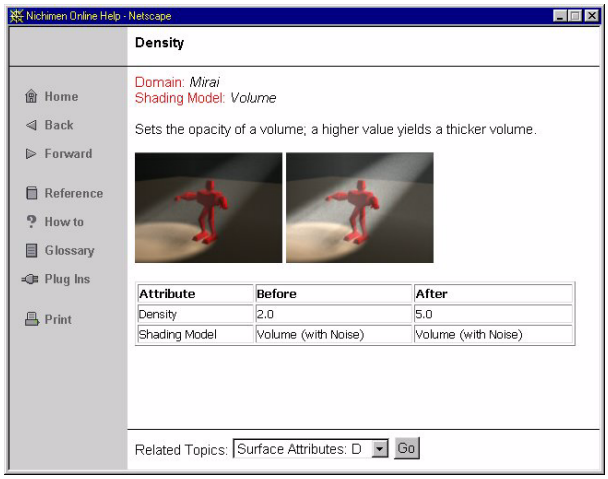
Image by Bay Raitt

When used in conjunction with volumetric effects and projection spot lights, truly spectacular effects can be achieved:

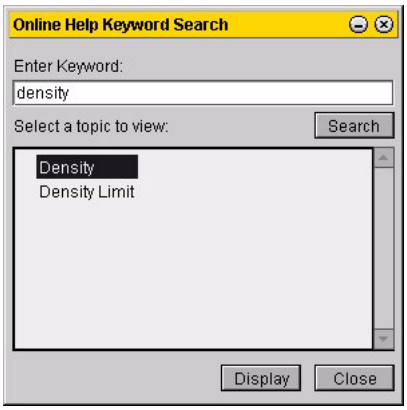
Image by Phil Zucco

ONLINE HELP

Mirai comes with a built-in on-line help system. You can get help for any screen element in Mirai by moving your mouse over the screen element and pressing *F1*; the related topic appears in a browser window:



You can also search the database by keyword if you know what you're looking for:

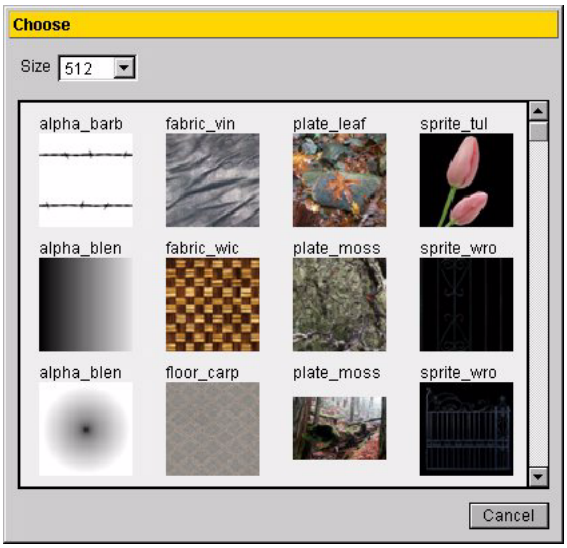


There are approximately two thousand help topics available in Mirai's online help system, including reference and tutorial materials.

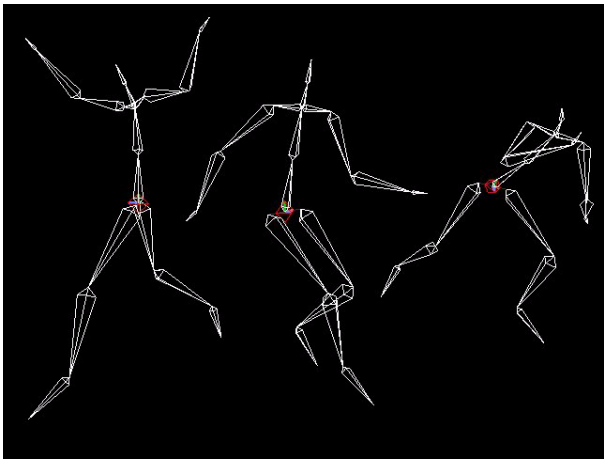
TEXTURE AND MOTION CAPTURE LIBRARIES

Mirai comes complete with two complete libraries to help you get started with your projects:

Mirai's texture library contains a wide variety of textures, including alpha maps, skin textures, fabric, plates, and more. All textures are supplied at true resolutions from 64 x 64 through 1024 x 1024.



Mirai also includes a great starter motion capture library, complete with a wide variety of motions supplied in Acclaim, BioVision, and Motion Analysis formats:



BUILDING A PIPELINE: GAME EXCHANGE 2.1

Game Exchange 2.1 provides an open, expandable, flexible, and highly customizable pipeline for importing and exporting data both to and from multiple platforms (both Mirai and Nendo have built in Game Exchange support). Users and interested vendors can write plug-ins which export data to Game Exchange's intermeditate file formats to facilitate data exchange between 3D packages.

Game Exchange is an open standard for describing 3D objects, textures, and animation in an easy to understand ASCII format—a format that can accommodate the data requirements of just about any 3D engine.

For developers working on projects which must be delivered on multiple platforms, Game Exchange 2.1 lets you generate content once, then convert that data to each of your target platforms, using different converters.

The Game Exchange API is written in C++, and has been completely revamped for version 2.1, making it simple for developers to create their own converters using a well-defined and well-documented API library.

To read more about Game Exchange, visit our web site:

- *http://www.wingededge.com/gamex/index.shtml*

MIRAI'S PLUG-IN ARCHITECTURE

An important part of Mirai's ability to fit into existing pipelines is the ability to create custom code via *plug-ins*. Plug-ins are "add-ons" which extend the functionality of Mirai for special applications; in fact, several internally developed plug-ins are distributed on the Mirai 1.1 CD. Third party plug-ins have been or are being developed by Testa-Rossa (motion capture editing), XO7 (audio tracks), and others.

Plug-ins can be automatically loaded at startup time, and the plug-in manager automatically detects and alerts the user to dependencies or inconsistencies between related plug-ins and Mirai.