

MTOR 基本貼圖

使用 MROT 建立一個 Planar-Z 投影貼圖



投影貼圖能加以變形、模糊，並且能持續的附著於模型表面之上。

投影貼圖能很輕易的將 2D 的貼圖貼附於 3D 模型之上，本節教學的方法，你也能很輕易的使用於別的模式及貼圖上。

針對於 Subdivision，投影貼圖亦提供了完整且容易的 2D 貼圖解決方案。

在這個章節，內容涵蓋了利用貼圖座標系統去設置 Z 平面(Planar-Z)投影貼圖的基礎用法。

接下來的步驟，同樣的又要利用 Slim 的樹狀系統，並在樹狀系統建立投影貼圖。



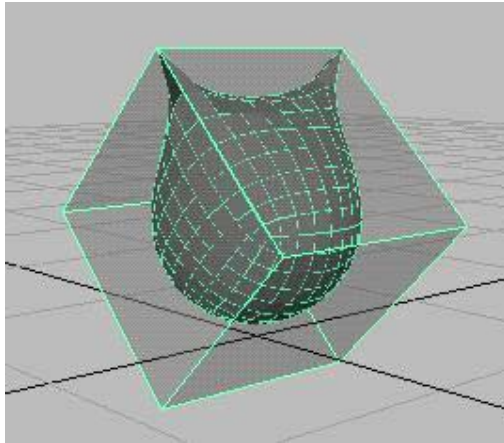
這是我們接下來要處理貼圖的模型檔

開啓範例檔

在這個範例，我們要為一個簡單的 Subdivision 物件施予貼圖，首先，先開啓範例檔：

mtor/ scenes/ kitty-projections/ kitty.ma

我們的目的地是為此模型貼上臉孔，在貼圖前，要先利用 MTOR 為模型建立一個投影貼圖座標系(*MTOR coordinate system*)。



這是要施予貼圖的模型，系利用一個 Cube 轉為 Subdivision 物件，並將上方兩個頂點設置為 Corner Hardness 而成。

譯註：

讀者若直接開啓 MTOR 的範例，有時會遇到無法 Render 並出現一行 Server Busy 的訊息，這是因為這個場景被開啓了網路算圖，因為沒有安裝網路算圖伺服器，因此便產生此一錯誤訊息，解決方法很簡單：

1. 執行 RenderMan -> RenderMan Globals...
2. 對話盒標籤切到 Spool -> Job Setup
3. 將 Renderer 的「netrender」改成「render」即可

譯註：

我們也可以依下列步驟來自行建立這一個 kit 模型。

1. 建立一個 Cube，並取名為「kit」。
2. 將「kit」Y 軸向上移動「0.5」的距離。
3. 將 X 軸旋轉「45°」的角度。
4. 執行「**RenderMan -> Pixar Subdivs -> Mesh as Subdiv**」，並將「**Subdiv Steps**」參數值設為「3」。
5. 選取 Cube 上方兩個頂點，執行「**RenderMan -> Pixar Subdivs -> Add Corners[]**」，將 Corner Hardness 設為「4」，如此耳朵部份才會有點圓圓的不會太尖銳，接著按下 Apply 執行。
6. 日後若要更改 Corner 的數值，需到 Hypegraph 裡，找到 mtorCorner 點選才能修改數值。

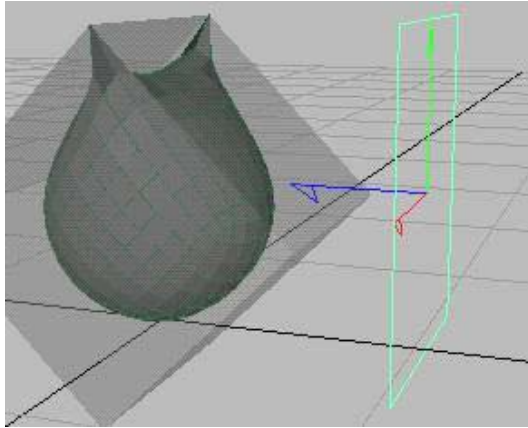
完成上述步驟 kit 即完成，但是，在 Maya4.0 中，利用 RenderMan 建立的 Subdivision 並不會被儲存起來，只要存檔後再開啓，模型便會消失不見，使用者必需更新到 Maya 4.01 以上的版本，或是利用 Maya 本身的 Subdivision 來完成此一練習，Maya 建立 kit 對讀者來說應該是輕而易舉的，不再贅述。

步驟 1 設定投影座標系統

首先，建立一個貼圖座標系統(以下使用一般人常用的貼圖軸稱之)：

Renderman-> New Coordinate System

執行指令後會建立一個帶箭號的可視貼圖軸平面，箭號的方向表示出貼圖軸為一個 Z 方向的投影貼圖，將此貼圖軸移動到模型的前方，如下圖。



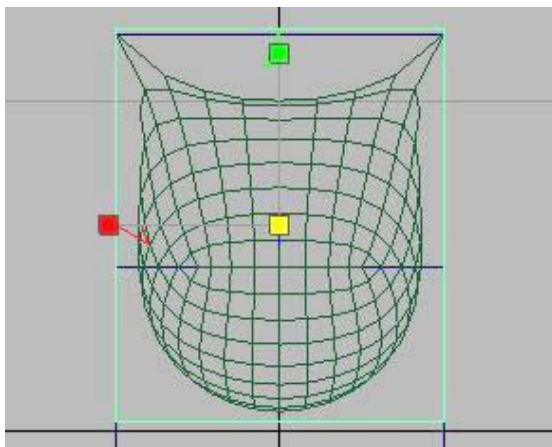
綠色的平面就是一個 Z 軸的貼圖軸

現在，要將貼圖軸調整到與模型相同的大小，此步驟需在 Front View 完成才能較為準確。

(在此例大約為 Y 軸 1.2 左右便能涵蓋整個模型)

步驟 2 更改貼圖軸名稱

通常預設的名稱都是沒什麼義意的字樣，現在，將貼圖軸的 SHAPES node 更名為「faceView」。



調整貼圖軸尺寸到模型大小，並更名為
faceView

步驟 3 匯入 Projection Shaders (投影 Shader)

在這個步驟中，要連結進來一個 Magic Surface 及一個 Magic Light，並要藉以去建立一個投影。

從 Slim 中匯入下 shader：

File-> Import Appearance

匯入的對話盒開啓，從 Go To 選單中選取「RAT Shaders」資料夾，找到「*magicSurf.slo*」及「*magicLight.slo*」這兩個 shader，匯進來後，Slim 結果如下圖。

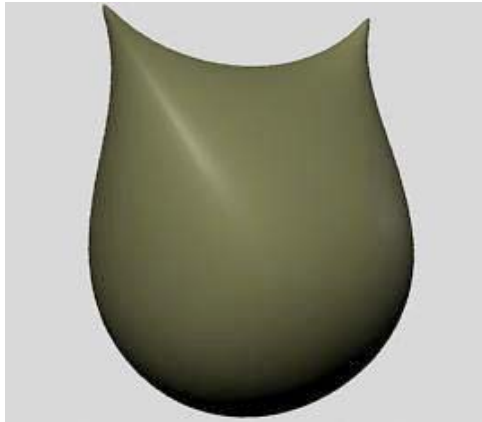


步驟 4 將 Magic Surface 附予給物體上

Magic Surface 是一個很特殊的 shader，可以幫助物體將貼圖透過 Magic Lights 投影到物體上。

在這裡，Magic Surface 要作為貼圖的基層，然後再附蓋多層的 Magic Lights 投影貼圖。

Magic Surface 的基本色為白色，將基本顏色更改為接近下圖的黃褐色，並附予給物體上。(顏色不需很精準，這並不影響這裡的學習)



步驟 5 設定 Magic Light

現在將 Magic Light 也 Attach 給物體上，我們要利用此一 shader 將「臉」貼給物體上。

現在雙按 Magic Light 的 shader 進入參數畫面，進入後首先要匯入臉的貼圖進來，從 Texture Name 欄位讀入以下檔案：

mtor/rmantex/kat.tif

接下來，必需把「.tif」轉換成「.tex」，這種格式是 Prman 唯一接受的格式，請執行 Texture Name 的圖形按鈕(在瀏覽檔案按鈕旁)，選取選單中的「**Convert-> Convert Texture**」，選取後，Texture Name 欄位中會多出一個「txmake」指令指示出.tif 轉換成.tex 格式。(圖形按鈕尚可設定貼圖的型式，詳細情形請參閱「*Working with Textures*」章節)

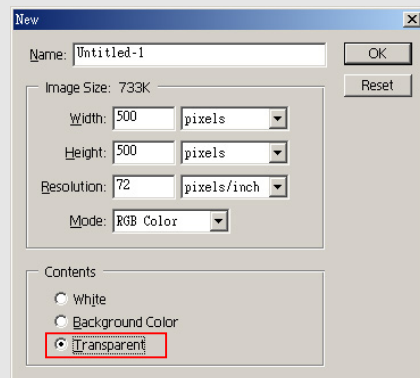
接著還需要開啓 Alpha Channel 來決定貼圖的透明部份，請將 Alpha Box 的核選方塊打勾以啓動 Alpha Channel，貼圖透明的部份會以 Magic Surface 來顯示。

譯註：

在 RenderMan 裡的 Alpha 與 Maya 本身的 Alpha 呼叫方式有些不同，大多的 3D

軟體都要另外讀入 Alpha 貼圖，但在 RenderMan 中，Alpha Channel 是介由 Tif 檔本身的 Alpha 來決定透明的，因此若貼圖要帶透明，需建立成透明的 Tif 檔，這是一種 32bit 的 Tif 檔，但與圖中帶 Alpha Channel 有點不一樣，透明部份並不透過 Alpha Channel，而是直接建立透明的底色，在 Photoshop 中無法直接存此種格式的圖，需透過以下步驟：

1. 開一個新的圖檔，並將背景色設為透明，如下圖。



2. 開出的新圖是為透明背景，直接在此背景上畫貼圖，透明部份即為我們不要的部份，如下圖。



3. 執行「File -> Jump to -> Adobe ImageReady」將圖匯至 ImageReady。
4. 從 Image Ready 中執行「File -> Export Original...」另存原始檔，並將檔案格式設成「Tiff」，儲存後即為透明的 Tif 檔。

步驟 6 Render

Render 場景，結果會類似下圖的樣子，但沒有 Bump。



這個結果是利用一個 MagicLight 建立貼圖，再利用一個 MagicLight 建立 Bump 的結果。

現在要再建立一個 Magic Light 來產生 bump，首先，先將 Magic Light 複製一份出來，從 Slim 中執行「*Appearance -> Duplicate -> Node*」，從 Texture Name 重新讀取檔案「*katBump.tif*」進來，當然的，必需執行 Convert Textures 轉換圖檔格式，請參考步驟 5，沒問題後，將 Magic Type 從「*Color*」改成「*Bump*」，最後將這個 Magic Light 再 Attach 到物體上。

好了，如此便完成了，讀者可以重新算圖，應該就會有 Bump 了。

製作精確的貼圖

到目前的練習都是依教學給的模型及貼圖來處理的，但我們實際要貼圖時，通常都要有一些依據來便於我們繪製正確的貼圖，以下的方法就是要來教各位如何產生一個參考用圖。

重新開啓 *kitty.ma* 並 Attach 好 Magic Surface，但不要連結 Magic Light 上去，讀者也可以將之前做的利用 Detach 指令將連結拿掉，最後確認場景中有 faceView

這個貼圖軸。

步驟 1 將攝影機設為貼圖軸

首先，開啓「RenderMan Globals」，並切到「Display」標籤，我們要將攝影設為貼圖軸視點，以便產生正確的投影貼圖，將「*Camera Name*」設為貼圖軸名稱，此例中名稱為「faceView」。

接下來設定貼圖尺寸，尺寸必需依據貼圖軸的比例來設，若是依教學中的貼圖軸為 1 : 1.2，建議設成 300 x 360。(這裡讀者需依自己的貼圖軸比例去設)。

重新 Render 場景，我們可以得到一個從貼圖軸看出去的正投影圖型，如下圖。



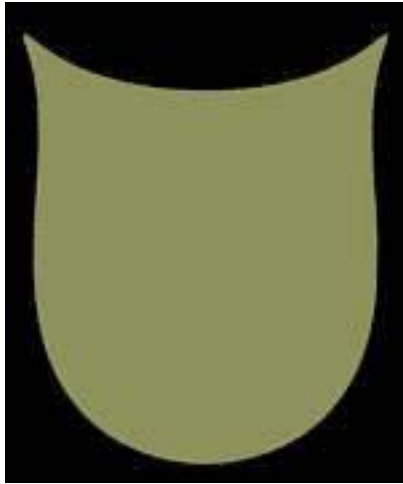
將貼圖軸設為攝影機所算出來的圖。

步驟 2 利用 Ambient Lighting 產生無陰影貼圖

要 Render 一張供繪製的貼圖，有時算出平面的圖會更加的方便，ambient lighting 便有助於我們製作這樣的一張貼圖，而且，算出來的顏色也能精確的與 Magic Surface 所設定的底色相吻合。

爲了不讓場景中原本的燈光影響產生的貼圖，先將場景中所有的燈光都 hide 起來，然後建立一盞 Ambient Light，並將 Intensity 設爲「1」，Ambient Shade 數值設爲「0」。

Render 場景，並將算出的影像儲存起來，結果如下圖，接下來我們就可以來繪製臉部了。



利用 Ambient light 算出的投影貼圖。

Render 後我們不再需要貼圖軸攝影機算圖了，請將 ambient light 隱藏或殺掉，將原本的燈光叫回來，然後將 Camera Name 清除，並將尺寸設爲 0 x 0。

步驟 3 利用 2D 軟體繪製貼圖

接下來利用 2D 繪圖軟體來繪製貼圖，並以 Alpha Channel 清除透明部份。

接下來建立 Bump 貼圖，黑色的部份爲平面部份，白色部份是凸起來的部份，儲存這些貼圖，並從 MTOR 讀入並貼到物體上。

譯註：

此 tiff 貼圖讀到 Photoshop 中，待要儲存時 Photoshop 會自動以 Save As...要使用者另存，此問題的解決方法請參考前面貼圖的步驟 5 裡面我寫的「譯註」部份。



color map

bump map



加入 alpha 透明的 color map

步驟 4 將貼圖貼到物體上

接下來，將自己畫好的貼圖貼物體上，參考前面步驟，利用 Magic Light 將 color map 及 bump map 均貼到物體上，並重新 Render。



摘要：

貼圖軸能幫助我們將 2D 的圖貼到 3D 的物體上，貼圖軸除了能投影 2D 的貼圖到物體上，尚能當做是攝影機，以利於 render 出貼圖的基底圖。



附錄 A 拉長的問題

當貼圖軸接近物體邊緣的時後，因為投影角度的問題，因此會產生貼圖被拉長的問題，解決此一問題，通常將貼圖拆開來，在不同的角度的面上使用不同的貼圖軸和 Magic Light 即可解決此一問題。



從放大的圖能清楚的看到因為像素減少產生的拉長現象

附錄 B 讓貼圖與物體一同變形(黏著 sticky)

有兩種方法讓貼圖黏著於物體之上，第一種方法為「凍結貼圖的頂點資訊」，先選取物體，執行：

Renderman > Vertex Variables -> _Pref: Freeze

此指令會建立一個存放貼圖位置資訊的節點(node)，接下來到 Magic Surface 裡面，將「sticky」參數打勾，如此貼圖便會跟隨著物件變形了。

第二種方法使用參考體(reference frame)，預設的 frame 為「1」，從 Magic

Surfshader 中將「sticky」以及「Output Reference Geometry」參數打勾，如此貼圖便會跟著物體一起變形。

譯註：

第二種方法我試了沒用，不過我都是用第一種方法，所以第二種方法並沒有深入去試，若哪位讀者有試過可以用，請 e-mail 給我。

e-mail : mkkii@ms49.hinet.net



貼圖黏著於物體之上一起變形

附錄 C 指定 Magic Collection Mode(投影模式)

到目前為止好像一切都完成了，但若是我們從物體的旁邊算圖，便會發現物體有兩個臉，這並不是我們所希望的。

現在打開 Magic Surface，找到「*Magic Collection Mode*」參數，改成「*Lights in Front*」，這項設定會使貼圖只貼到貼圖軸對應的那個面上。

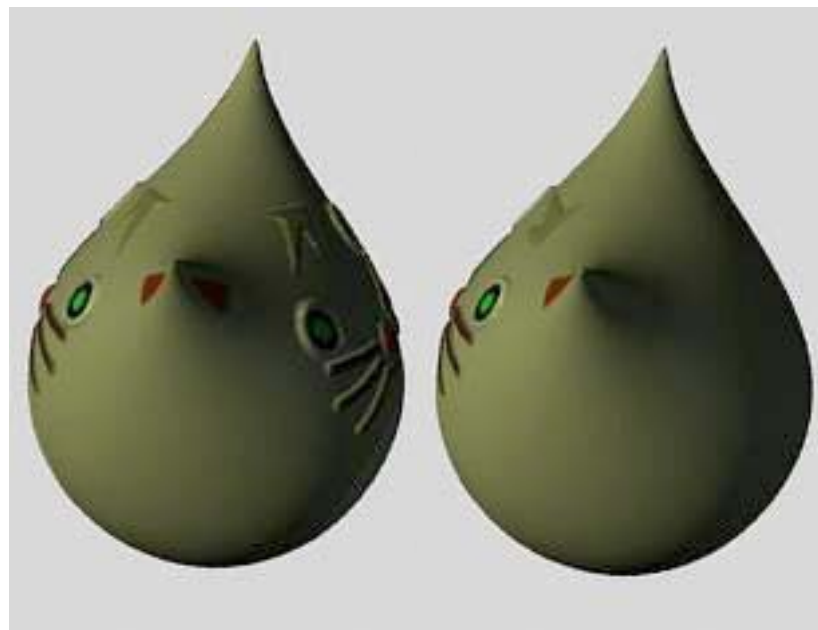
這裡有個附屬的設定，就是 Shadow Map 的設定，設定 Shadow Map 對於一些複雜的多層貼圖，只有第一次會多花些時間，後來更換 Magic Light(即貼圖)但不需再次的去運算其投射影子。

譯註：

這裡教學文件有寫錯，要設置 Shadow Map 的是 Light Shader，而不是 Magic Light，讀者若要設置影子，請再從 Slim 中執行：

File-> Import Appearance

從對話盒中的 Go To 選單中選取「RAT Shaders」資料夾，找到「mTORSpotLight.slo」匯入，對附予給燈光，其他請參照我上一份翻譯「MTOR Tutorials」的 shadow map 設定。

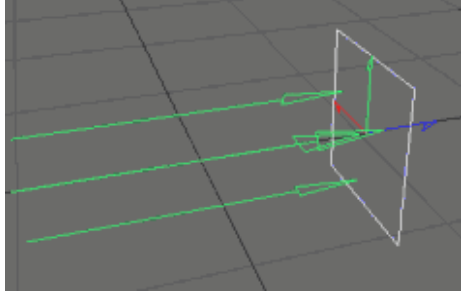


All Light

Lights In Front

附錄 D 關於貼圖軸的對齊

有時貼圖軸無法在標準的三個投影平面對齊，此時就需建立正投影平面來輔助調整貼圖軸，這一部份請參考章節：[建立貼圖軸視點](#)。



貼圖軸視點

譯註：

整個練習完，讀者可能會覺得自己做出來的與教學圖片有一個很大的不同，就是設定 bump 之後，我們的模型只有看起來只有正面有凹凸，轉到邊緣凹凸就不見了，請看下面兩張圖：



邊緣部份有凹凸

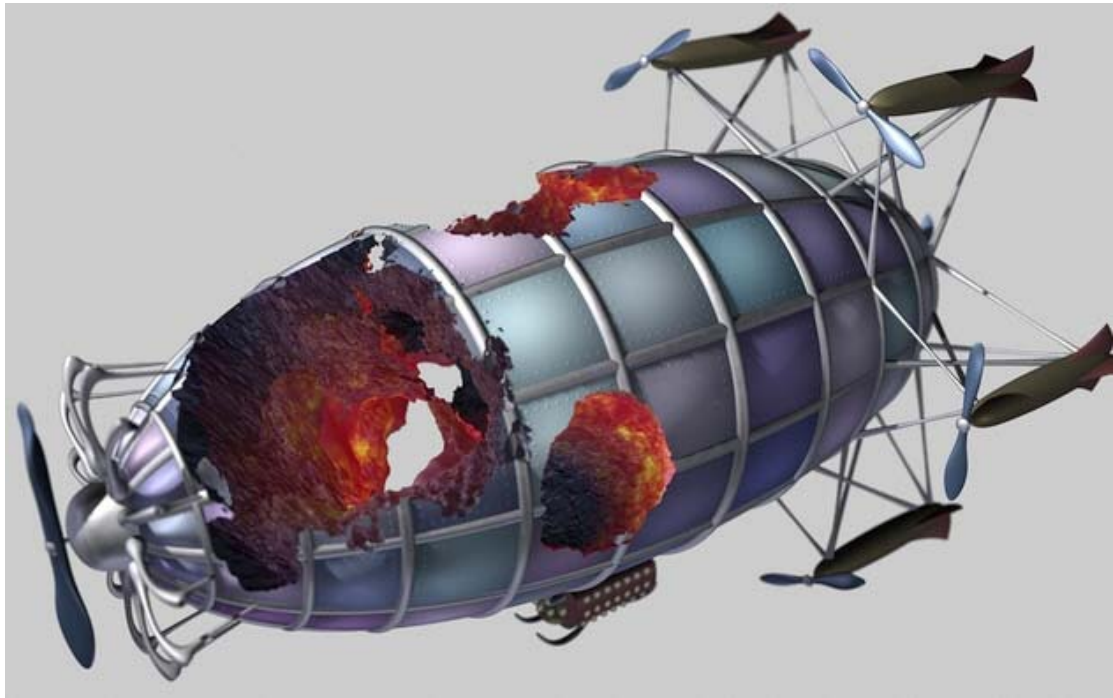


邊緣部份很平滑，沒有實際的凹凸

這是 Displacement 的設置，需指定要 displacement 才会有真正的凹凸，Maya 本身也是這樣的，在這裡要設置 displacement 很簡單，只要打開 Magic Surface，勾選裡面的「Displace Surface」這個參數即可。

使用 CSG

MTOR 提供了一種特殊的表面布林運算效果，CSG 是「Constructive Solid Geometry」的簡稱，可讓多個物件(Object 或 Group)相互結合運算，產生特殊效果。

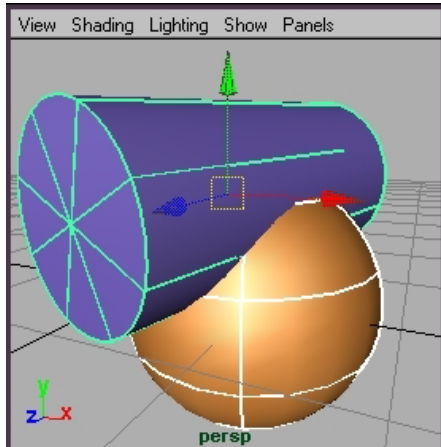


利用 CSG 產生出來的飛船效果

要使用 CSG 必需將物體指定為 CSG primitives，然後將其群組起來，再以 CSG 指令加以運算。

1) 建立 CSG primitives

首先在 Maya 中建立兩個樣型，如下圖(最好是封閉的實體模型)。



接下來要指定為 CSG 物件，選取物件後，從 RenderMan 的選單中執行：

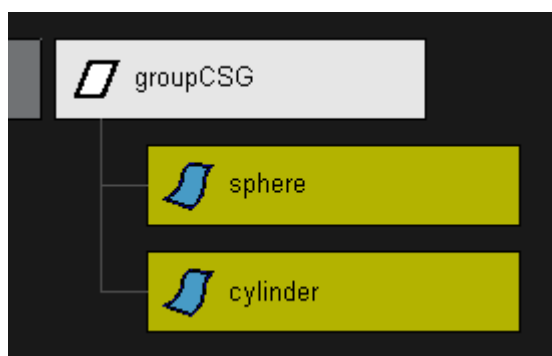
Attributes-> CSG-> Primitve

這兩個物件被指定成為 CSG 物件。

2) 將兩個 CSG 物件群組起來

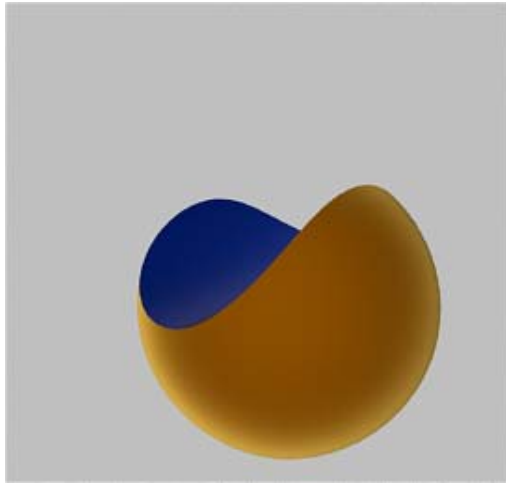
選取兩個物件並群組起來(Ctrl + g)，要利用 Group 物件來決定運算的方式，這裡要用球減去圓柱體，選取 Group 物件並執行。

Attributes-> CSG -> Difference



3) Render 相減後的結果

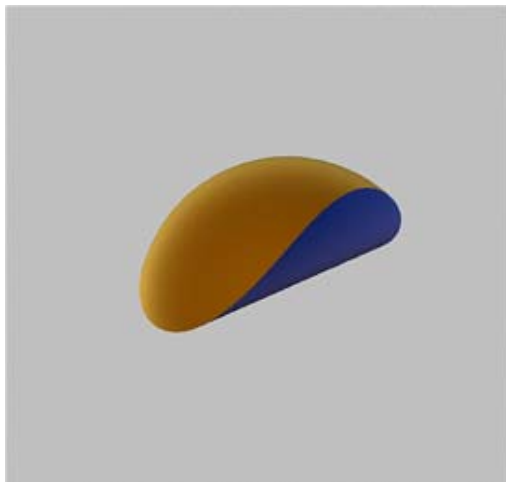
直接執行 RenderMan 的 Render 算圖，可得如下圖的結果。



若讀者算出來的圖相反，是球減去圓柱，那是順序的問題，物件的前後順序問題，若算出來是相反，只要在 Outliner 將球體以滑鼠中鍵往上拖曳，到圓柱體上方即可，也就是說，物件的高低順序決定物體相運算的順序。

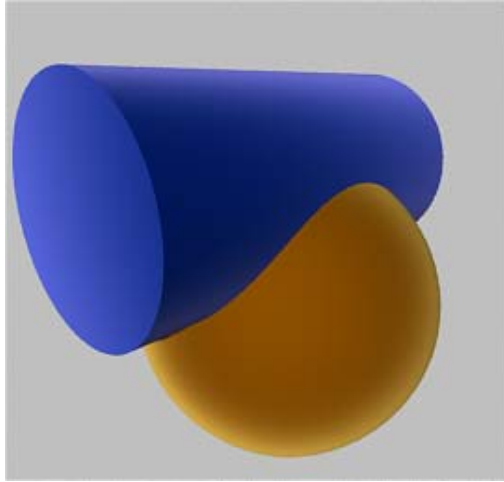
4) 交集運算的結果

將 group 改指定為 intersection，「Attributes-> CSG-> Primitve」，算圖結果如下。



5) 聯集運算結果

這次將 Group 改為「**Union**」，會算出如下結果。



6) 結論

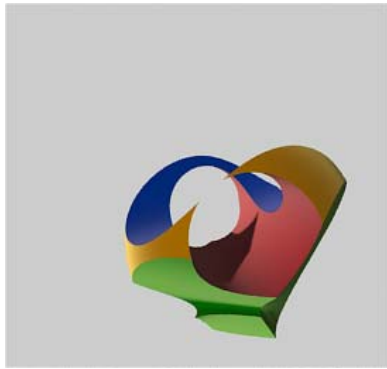
CSG 是一個功能強大的運算方式，CSG 可以同時對多個物件及其材質做布林運算，以下為幾個重點。

- 將要 CSG 運算的物件群組在一起，並需確認要運算的物件均被指定為 CSG primitives。
- 能很容易的製作動畫，並支援 Motion Blur。
- 透過 Group 的 CSG 設定，能做多種運算。
- 支援多層式運算，即 Group 裡面還可以有 Group，每一層的 Group 均可設定為不同的交集、聯集、或是相減的運算。

譯註：

CSG 運算連貼圖都能正確的被加以運算，且亦支援 displacement，請讀者自行測試。

下圖即是一個多種運算結合起來的結果。

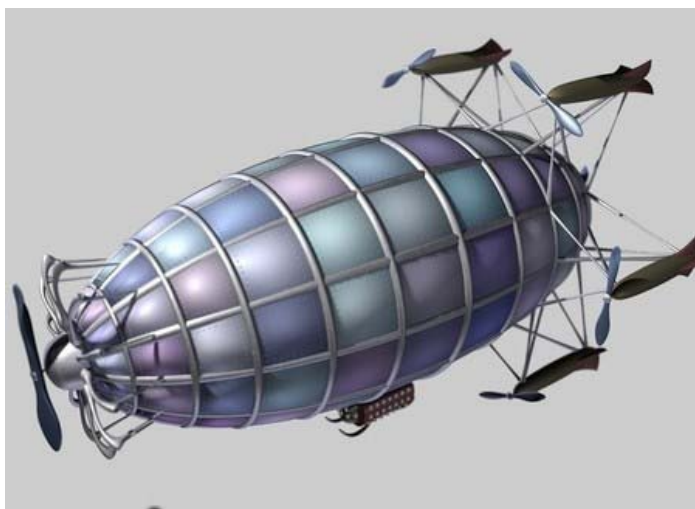


被炸毀的飛船實例研究

我們可以利用 CSG 來製造出一艘飛船被炸毀的情形，建造一艘飛船後，再利用 Displacement 創造幾個挖洞的物件，加以相減運算後，便可做出飛船爆炸破壞的感覺。

飛船的處理

將飛船的所有物件群組後，設為 CSG 物件。



挖洞用的模型

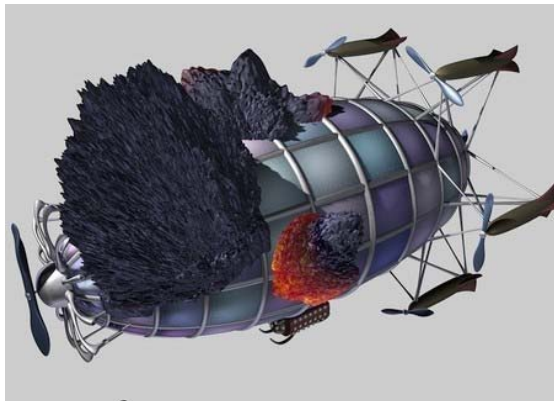
創建三個挖洞用的模型，這三個模型是利用球體變形後，再附予 displacement shader 而做成的。

這三個模型也設為 CSG 物件。



將挖洞物件擺放到正確位置

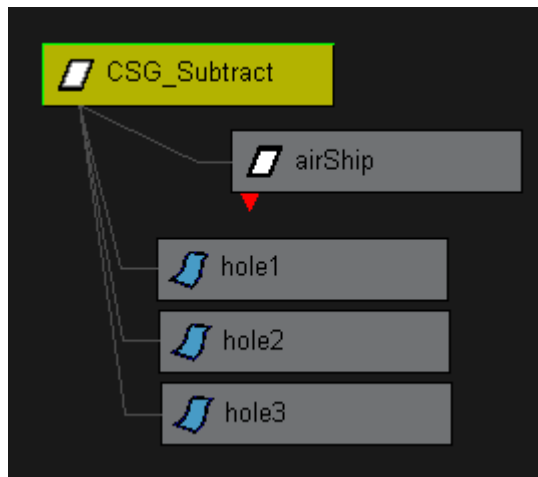
將挖洞用的模型放到如下圖的位置，一半陷入飛船裡面。



將所有模型群組起來

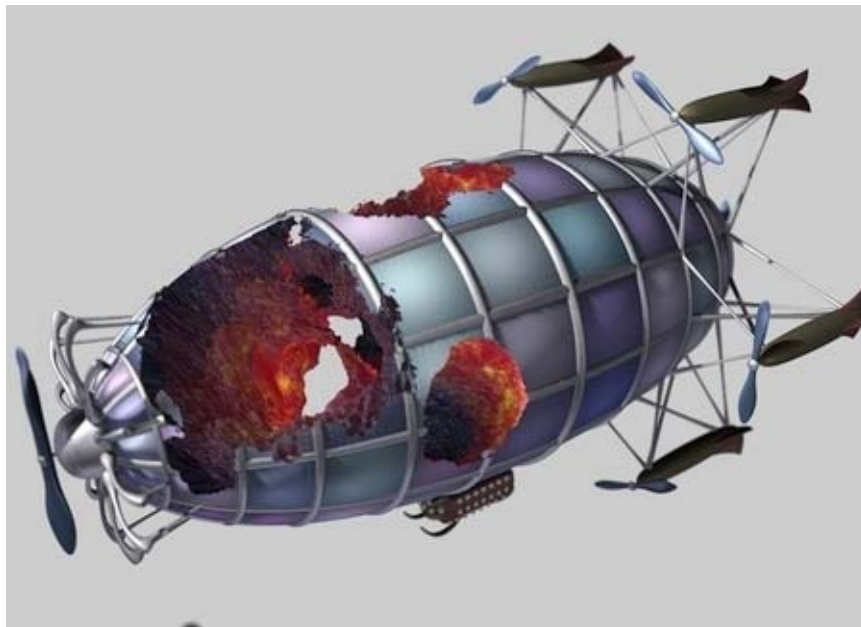
最後將所有模型群組起來，並對 group 設為相減的 CSG Difference 運算。

因為有運算的前後關係，故需將飛船擺在第一位，其他的挖洞物件放至後面，如下圖的 group 結構。



最後結果

最後算出來的結果如下圖。



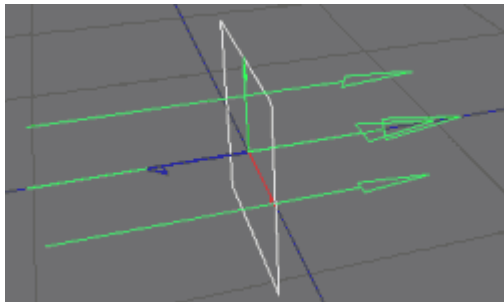
建立貼圖軸視點(Coordinate Viewer)

在許多的場合，一個物件會用到多個貼圖軸，而且這些貼圖軸也不會固定在三個標準投影視圖上，這時，建立貼圖軸視點便是一個很有用的工能了，他可以為我們做到以下兩個重要的好處：

1. 能更容易的放置並調整貼圖軸。
2. 能更精確的繪製所需的貼圖。

步驟一

建立一個平行光(direct light)，並將光源關閉。

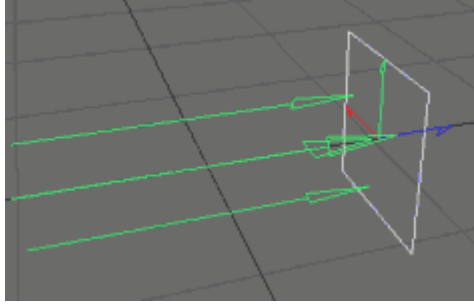


一個平行光和一個貼圖軸，貼圖軸的方向(藍色箭頭)與平行光的方向是相反的。

步驟二

將貼圖軸轉 180 度後，沿 Y 軸移動到平行光的箭頭頂端。

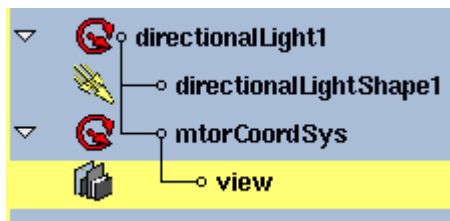
貼圖軸的藍色箭頭與平行光的箭頭必需同一方向。



透過平行光視點，我們便可以取得一個貼圖軸的檢視點。

步驟三

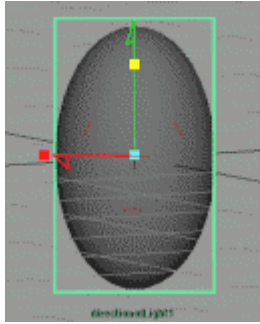
將貼圖軸設為平行光的子物件，可以從 HyperGraph 或是 Outliner 直接以滑鼠中鍵拖曳即可，亦或是利用 Parent 指令亦可。



將貼圖軸設為平行光的子物件，此例的貼圖軸名稱為「view」，若要貼圖時，Magic Light 亦要對應到 view 才行。

步驟四

選取平行光物件，並從 View 視窗執行「Panels -> Look Through Selected」。



平行光視點是一個正投影平面視點，因此能精確的對準貼圖軸，上圖中綠色框便是貼圖軸，選取後再調整到所需的大小，若是模型超出螢幕太多，可按「f」自動將 View 移到全畫面。

結論

建立貼圖軸視點是一個很有用且簡單的功能，透過平行光我很很容易的便得到一個正投影平面的視點，再利用此正投影視點，我們便能輕易的來調整貼圖軸的大小及位置。

譯者：M.K.

E-mail：mkii@ms49.hinet.net