Viewpoint

Authoring Viewpoint ImageLayer Content

Version 1.0
February 2003

Authoring Viewpoint ImageLayer Content

Author: Caleb Hill

Contributors: Ryan Case

Viewpoint Corporation
498 Seventh Avenue
Suite 1810
New York, NY 10018

Contents

Chapter 1: About ImageLayer

This document describes how to utilize the Viewpoint-proprietary ImageLayer authoring technique to render 2D product display images online in various color and pattern options.

> **Important:** The information provided in this document is intended for content developers who are familiar with industry standard design and 3D modeling tools. ImageLayer content authoring is best suited for the Windows platform.

This document includes the following topics:

- Chapter 1: "About ImageLayer" — Explains the Viewpoint technology platform, file structure, software and system requirements, and describes the advantages of the ImageLayer content.
- Chapter 2: "Choosing ImageLayer Assets" — Provides help in choosing assets that best lend themselves to the advantages of ImageLayer content.
- Chapter 3: "Using ImageLayer Techniques" — Describes how to transform 2D product display images into ImageLayer content, including color swapping and pattern swapping.
- Appendix A: "Help, Resources, and Feedback" — Lists Viewpoint resources available to you.
- Appendix B: "Sample MTX Control Files" — Provides two sample .mtx files, one illustrating a color swapping image, the other depicting a pattern swapping scene.
- Appendix C: "ImageLayer XML Elements" — Describes the Viewpoint XML tags you can use to convert your product display images into ImageLayer content.
- "Glossary" — Defines the terms used in this guide.

Viewpoint Media Files

A Viewpoint scene includes the following proprietary Viewpoint media files. Most 3D authoring applications export the Viewpoint .mts and .mtx (.mtz) 3D scene files.

- .mts file — Contains a compressed collection of rich media components. These components are orchestrated by the .mtx file to create a scene.
- .myx file — An XML-based file containing the hierarchical relationships between elements in the scene; also serves as the script for staging the elements. This file instructs Viewpoint Media Player how to handle the rich media components it finds in the related .mts files for the scene. These instructions include information about position, rotation, scale, opacity, and so on. This file can also reference external files (such as Flash .swf files), separate .mts files, and even other .mtx files.

  > **Note:** Broadcast Key files also use the .mtx filename extension and are typically named BroadcastKey.mtx or bkey.mtx.

- .mtz file The binary compressed version of an .mtx file.

- .mzv file A file format for compressed image tiles (parts of a large image) used by the Viewpoint ZoomView technology, a component of Viewpoint Media Player.

> **Note:** Even though a rich media component is in an .mts file, it is not necessary to use it in the resulting scene. The .mts file is your creative arsenal. The .mtx (.mtz) file contains instructions for acquiring and arranging rich media resource files from the compressed .mts file. You can also store media resource files (such as Flash .swf files, sound files, and JPEG textures) outside a scene. The .mtx file refers to these files and applies them to the scene.

Viewpoint Corporation offers tools and help to bring your 3D and rich media content to the web. To find out more about the Viewpoint family of applications and utilities, visit Viewpoint Developer Central at http://developer.viewpoint.com/.

About ImageLayer Content

The ideal content for your online retail store, ImageLayer content can display in multiple color and pattern options, providing exceptional retail opportunities for such product lines as clothing, accessories, furniture, interior design, and ceramics.

Viewpoint ImageLayer enables your online retail store to drive higher revenue through:

- Increased customer access to all available product options and accessories.
- Improved customer satisfaction, brand perception, and brand loyalty.
- Reduced costs associated with creating and storing multiple product images.

Increasing Access to Your Product Options and Accessories

By leveraging Viewpoint ImageLayer, you can enable one master product display image with color or pattern "swapping" functionality. "Swapping" functionality provides your customers the option to preview your products in all available colors and patterns. For example, ImageLayer's unique color swapping technique drives sales for all available color options for a given product, not just those featured on the product model.

Satisfying Your Customers and Building Your Brand

Viewpoint ImageLayer provides a shopping experience that improves brand perception and increases sales by serving photo-quality product images in real-time.

Reducing Costs

ImageLayer content authoring decreases the cost associated with creating and storing a product collection online by generating all available product display images from a single master image.

ImageLayer Technical Overview

Viewpoint technologies operate in a serverless capacity, meaning they require only standard HTTP servers for deployment. Viewpoint's "serverless" operation translates into:

- Easier setup, better scalability, and more cost effective maintenance.
- Greater flexibility from a server/IT perspective.

- Reduced server load, as the server is being used only as a delivery platform and not as the actual compositor.

Additionally, ImageLayer content leverages the "standard" strengths of the Viewpoint graphics operating system. For example:

- Viewpoint client-side software reduces server load, making your server more responsive.
- The Viewpoint graphics operating system does not depend on Java or other third-party technology.
- Viewpoint supports multiple media types, such as 2D and 3D images and animations.
- The Viewpoint XML control files (MTX) are easy to re-configure and easy to integrate into existing web pages.

ImageLayer Architecture

ImageLayer authoring techniques include a complete image swapping ("compositing") architecture, the foundation of color and pattern swapping, enabling you to:

- Create swappable (alpha-composited) images with industry-standard authoring tools.

  Note: Both the color and pattern swapping processes use 3D assets, which enhance image rendering, but do not incorporate traditional 3D models.

- Activate, position, and configure your swappable images with simple XML commands.
- Build interactivity into your images.
- Trigger interactivity within your images via external HTML controls and standard DHTML scripting.

Color Swapping

Color Swapping uses texture color modulation, which is equivalent to Photoshop's color multiplier, to create color variations for a product display image. By defining color areas onto the underlying geometry on which an image is mapped, the color of specific areas of an image can be changed by animating the diffuse color of the geometry.

Pattern Swapping

Pattern Swapping uses UV meshes, mapped with swappable tiled textures, shaped to match a surface in a product display image. In this way, the UV meshed areas in the image are enabled to be viewed in various patterns.

Software and System Requirements

Creating ImageLayer content requires the following software and systems capabilities.

Required Software Applications

- Adobe® Photoshop™ or Illustrator™
- Any 3D modeling, animation, and rendering software
  - 3ds max™ by discreet® is Viewpoint's preferred software for the Windows platform

  - Carrera Studio™ by Eovia® is Viewpoint's preferred cross-platform software
- Viewpoint Scene Builder
- Viewpoint Media Player
- Netscape Navigator 7 or later, Safari 1.1, or Microsoft Internet Explorer 5.x or later
- Any XML text editor
  - XML Spy® is Viewpoint's preferred editor for the Windows platform

Remember: To download Viewpoint Scene Builder and Viewpoint Media Player, go to Viewpoint Developer Central.

Minimum System Requirements for Viewpoint Tools

Viewpoint Scene Builder for Windows

- Microsoft® Windows® 98, Windows 2000, Windows Millennium Edition, Windows NT® 4.x, or Windows XP
- Pentium® II 300 MHz processor
- 128 MB RAM

Viewpoint Media Player

Windows Operating System	Macintosh Operating System
• Microsoft® Windows® 95, Windows 98, Windows 2000, Windows Millennium Edition, Windows NT® 4.x, or Windows XP • Netscape Navigator® 7.x , Microsoft® Internet Explorer 5.x or AOL 7.0 • 64MB RAM • Pentium® II 300 MHz processor	• Apple® Macintosh® Jaguar ™ 10.2 • Safari ™ 1.1 or Microsoft® Internet Explorer 5.2 • 256 MB RAM • PowerPC G3®

Chapter 2: Choosing ImageLayer Assets

ImageLayer content provides real-time, photo-quality image displays composed of the following assets:

- Bitmap Graphics — One 2D image (a JPEG or PNG file) is all you need to get started. From this one image, you can display your entire inventory for a product line, such as furniture pieces, wall hangings, and handbags.
- Multiple swappable products or areas — Within each product image, you can create multiple color and pattern swappable areas.
- Multiple color and pattern swatches — You can create multiple color and texture options for a given product. Customers access these options by clicking thumbnail images that display alongside the product.
- MTX control file — The .mtx file describes the product display information and controls how the product renders.

Choosing the Right Product Assets for Display

Consider the type of product you sell as the starting point for understanding how ImageLayer techniques can work for you. For example, a color-swappable handbag can be very simple to implement, while enabling an entire furniture display for pattern swapping requires much more modeling and shadowing work.

When choosing the assets you intend to enable with ImageLayer techniques, consider:

- The number of objects in a scene
- The type of surfaces inherent to a scene's objects
- The amount of shadowing, contouring, color, texture, and detail in a scene's objects

Number of Objects

To determine an ImageLayer project's complexity, first consider the number of swappable objects you want in your product display image. Color and pattern swapping requires 3D geometry Photoshop masking, a labor-intensive process. As you increase the number of swappable objects within a product display image, it becomes more difficult to maintain cohesiveness between the swappable objects in a scene.

Types of Object Surfaces

There are two types of swappable object surfaces:

- Flat surfaces — Product display images that feature flat objects lend themselves to simple color and pattern swapping implementation. For example, wallpaper swaps really easily because walls tend to be flat surfaces.
- Non-flat surfaces — When an object contains grooves and contour changes, it becomes a non-flat surface. Non-flat surfaces stretch patterns and fabrics, breaking the pattern and causing inconsistencies within the fabric. To add patterns and texture changes to non-flat surfaces, you must ensure that the pattern and fabric is consistently applied across the surfaces.

Shadowing and Masking Contours

Objects with crevices, curves, contours, and finite detail require far more complex shadowing and masking than simpler objects. For ImageLayer projects, complex objects require you to strip the object models you create for swapping of most of their color fabric and textures.

For example, in the illustration below, the chair on the left, with its punched buttons, intricate detail, and heavily contoured surfaces, exemplifies a complex object. The chair on the right, with its relatively flat, light-colored, and detail-less surfaces, represents a fairly simple object. When making these objects swappable, the chair on the left requires far more shadowing and complex masking than the chair on the right.

Note: When an object has contour grooves, the model must be appropriately shadowed and contoured to look realistic when colors or pattern swatches are applied. The more grooves and contours present on an object, the more time you must spend modeling to create shadow and contour effects.

Chapter 3: Using ImageLayer Techniques

This chapter provides steps for creating swappable product display images with ImageLayer content authoring techniques.

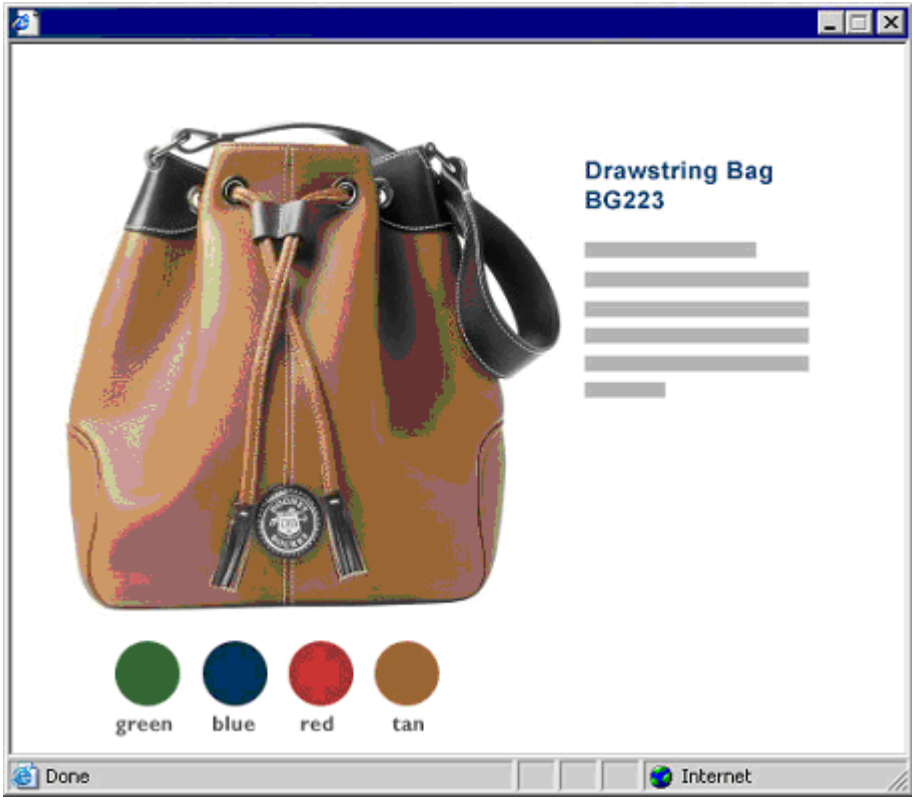
> **Important:** This chapter provides sample illustrations and step-by-step instructions describing how to complete ImageLayer projects. However, these tools and processes may not apply to every ImageLayer project; you may find other tools or processes that work better for your needs.

Color Swapping

ImageLayer color swapping enables your online customers to preview a product in different colors. When your customers select a new color option from the available color swatches, the new color replaces the existing display color.

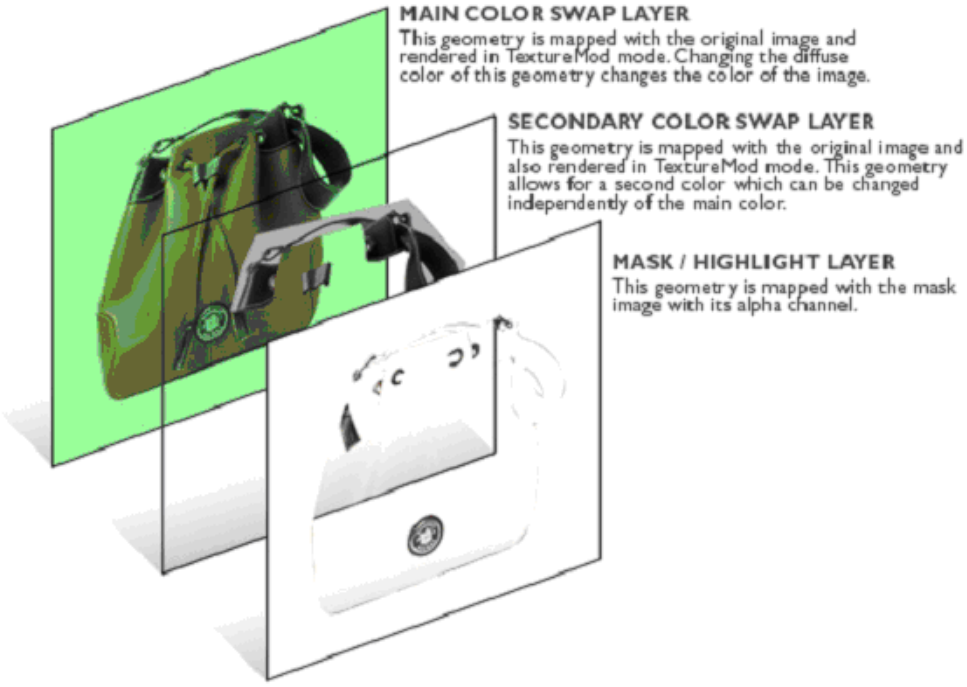
For example, the following illustration provides a typical market application of color swapping. This product display image supports four swappable preview colors that correspond to two color areas of the bag (main color and accent color). It is possible to enable the image with infinite color swatch options and multiple color areas.

> **Note:** Throughout the rest of this section, we will use a handbag example to explain how to enable your product display images with color swapping functionality.



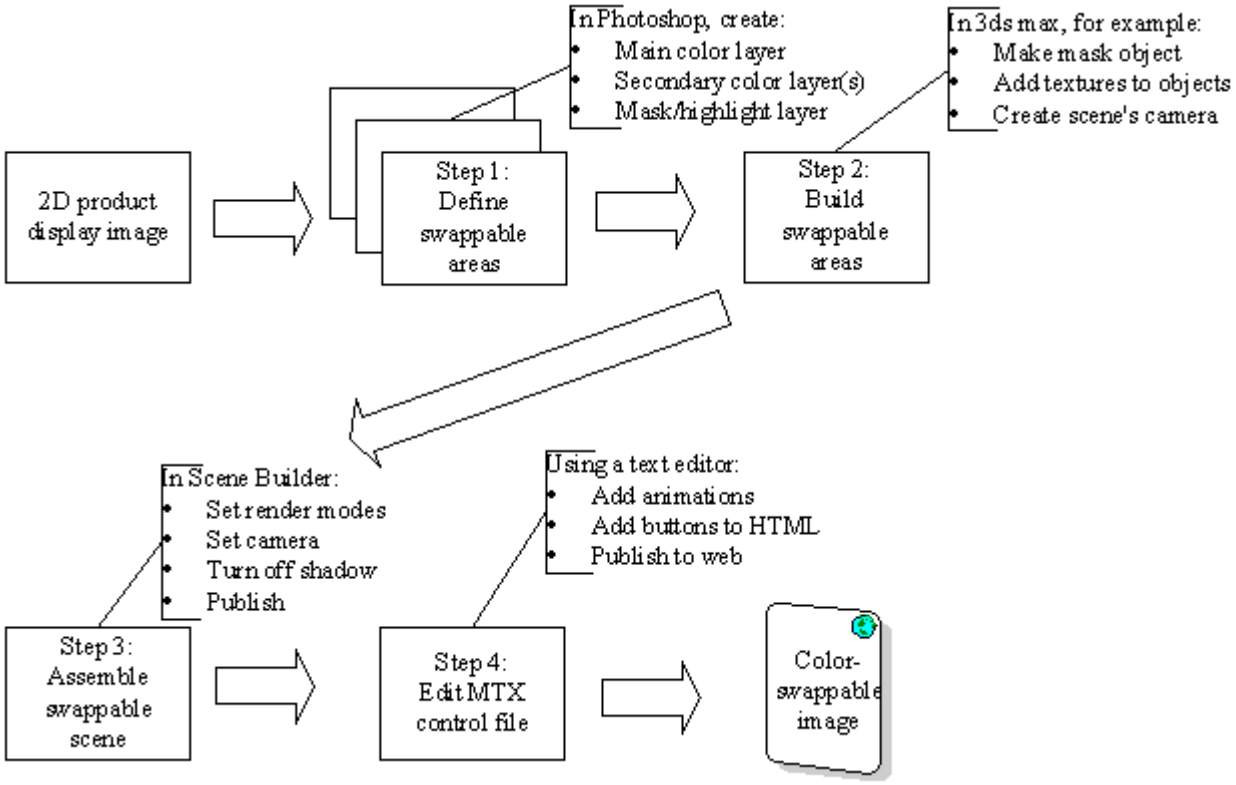
Sample Color Swapping Project

The following illustration provides a high-level description of our handbag project. The image of the bag has been separated into three sections, each illustrating a component for our color swapping project.



Workflow

The following graphic illustrates the workflow involved in creating the color swappable handbag. Follow these steps to create your own swappable product display images.



Tip: For help choosing product display assets that are most compatible for color swapping, see Chapter 2: "Choosing ImageLayer Assets".

Step 1: Defining Swappable Areas in Photoshop

Step one of the color swapping process requires you to use Photoshop to mask the unswappable areas of your product display image and to define the highlights of these areas.

Process Overview

The following steps present the high-level Photoshop design process for enabling your product display images with color swapping functionality. Each step is explained in more detail below.

1. "Opening and saving your image in Photoshop" — Saving your image as a .psd file gets you started!
2. "Masking the nonswappable areas of your image" — Masking protects the areas of your image that your online customers cannot swap. These areas include latches, rivets, and so forth.
3. "Extracting highlights from your image:" — The highlight areas refer to the white highlight reflections of the bag in the following sample illustration. It is necessary to extract these highlights from the image because the multiply effect

used to create the bag's color variations removes or lightens these highlights. Creating a layer specifically for the highlights keeps them intact, preserving the reflective quality of your image.

4 "Building a shape for a secondary area of color swapping (optional)" — You can define as many swappable color areas for your product display image as you desire. Each secondary color area in your image must be defined as described in the steps below.

5 "Creating a solid fill layer" — This layer provides the color values that, when multiplied through the original image, constitute the image's color variations.

6 "Previewing color variations"— This step allows you to ensure the color variations match your requirements.

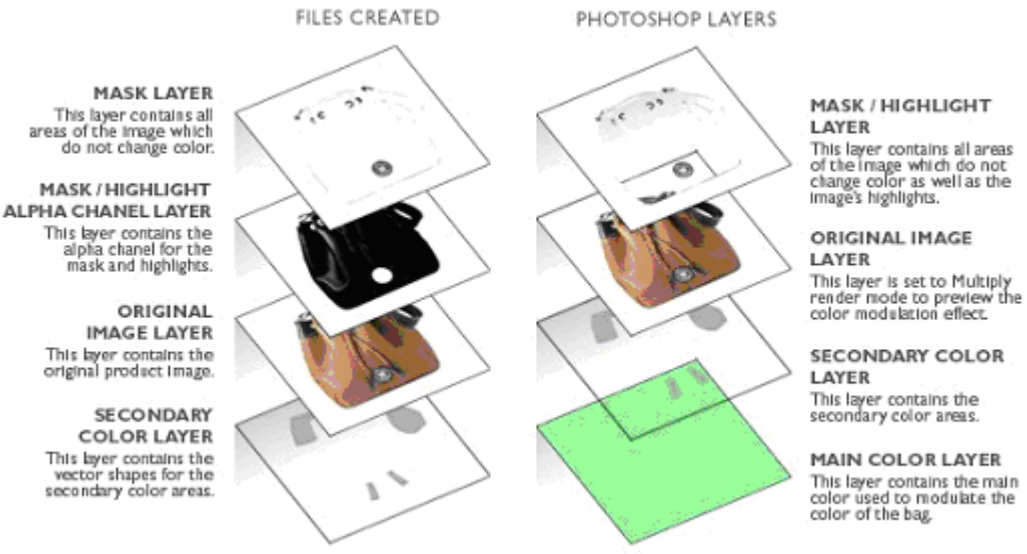
7 "Create a mask alpha channel layer" — The mask alpha channel layer enables you to add levels of transparency to your mask layer.

8 "Creating a flat lightmap"— The flat lightmap ensures that no lighting effects are added to the image surface.

9 "Exporting the Photoshop layers" — Exporting your image layers allows you to begin the modeling process.

Sample Photoshop Illustration

The following illustration exemplifies the first step of the ImageLayer color swapping process, including the export layers and the color preview layers. In the example on the left, we have created four Photoshop layers to define the files we created to build the swappable areas of the product display image (the bag). In the example on the right, we have ordered the layers we created to preview the colors variations for our product display image (the bag).



Step-by-Step Instructions

In our sample color swapping project, we defined the swappable areas of the handbag using the following procedures. Depending on your project requirements, these steps may or may not apply.

Opening and saving your image in Photoshop

1. Open Adobe Photoshop.
2. Go to the **File** menu; select **Open** and browse to select the product display image to open.
3. Save your image as a .psd file. For example, this area is described in the Photoshop sample illustration as the Original Image layer.

Masking the nonswappable areas of your image

1. Use the **Pen** tool to trace (mask) the nonswappable areas of the image (the image areas where you do not want to change color, including any white space around the image).

   Tip: The Pen tool automatically creates a separate layer for your masked shapes. You can use a variety of tools in addition to the **Pen** tool to mask the nonswappable areas, such as the **Magic Wand** selection and the **Brush** tool.

2. Select your mask layer.

   Tip: To select the mask layer, position the cursor over the layer on the Layer Palette and hold down the **Control** button on your keyboard while clicking the mouse.

3. Go to the **Edit** menu and select **Copy Merged**.
4. Go to the **Edit** menu and select **Paste** to create the mask layer. For example, this area is described in the Photoshop sample illustration as the Mask layer.

   Note: Depending on the masking process you used for your project, you may have additional layers than those named in these instructions. You can delete these once you create the mask layer.

Extracting highlights from your image:

1. Go to the **Select** menu and choose **Color Range**.
2. From the **Color Range** dialog box, define your image's highlight areas by clicking a representative highlight area from your product display image.
3. Adjust the **Fuzziness** slider to create the selection area you desire and click **OK**.
4. Select the **Layer** menu and create a new layer to contain the image's highlight areas.
5. Go to the **Edit** menu and select **Fill**.
6. Fill your selection area with the color white. This layer, which will be merged with the Mask layer, is described in the Photoshop sample illustration as the Mask/ Highlight layer.

   Tip: Adjust the layer's opacity to alter your highlight intensity.

Building a shape for a secondary area of color swapping (optional)

1. Select the **Layer** menu and create a new layer for the secondary color area. For example, in the Photoshop sample illustration, we have described this layer as the Secondary Color layer.
2. Trace (mask) the image's secondary swappable area (the image areas where you want to change color in addition to the main image).

   Tip: Use the **Pen** tool to fit these shapes precisely along the edges adjacent to the image's swappable areas, but don't spend too much time fitting them shape precisely over the non-swappable areas because the Mask layer, described above, provides these areas a clean edge.

   Tip: When masking the secondary swappable areas, adjust the opacity of your shapes to better view the areas you are masking.
3. Go to the **Tools** palette and select the **Shapes** menu. Apply a rectangle that covers the entire layer. This rectangle is used to align the secondary color area with the original image.

   Note: In the following two steps, you will combine all of the shape layers you use to mask the secondary color areas into one shapes layer, thereby facilitating their export to the Illustrator (.ai) file format.
4. Select the rectangle layer and use the **Direct Selection** tool to copy the shape into your shape layer.
5. Select the other shape layer and paste the rectangle into it to link the shape layers into one layer.
6. Repeat steps 4 and 5 for any additional shape layers.

Creating a solid fill layer

1. Go to the **Layer** menu and click **New Fill Layer**.
2. From the **New Layer** dialog box, give the layer a unique name and select the fill color of your choice. For example, in the Photoshop sample illustration, we have described this layer as the Main Color layer.
3. Click **OK**.

Previewing color variations

1. Arrange your image's layers as follows, from top to bottom, to preview your color variations:
   - Highlight layer
   - Mask layer

     Note: For preview purposes, mask and highlight layers can be combined.
   - Original Image layer

     Note: This layer's color is the color that is multiplied (or modulated) with the original image to create a new color.

- Secondary Color layer(s)

  Note: Secondary Color layers are included only if you have multiple color swappable areas in your product display image.

  Tip: For preview purposes, duplicate this layer and remove the rectangle from the layer you use to view color variations. When done previewing your color variations, delete the rectangle-less duplicate layer.

- Main Color layer

2. Select the Original Image layer.
3. From the Original Image layer, set the render mode to Multiply, which multiplies the colors of the underlying layers into the image, causing the image's color to change. For example, in the Photoshop sample illustration, the original bag's color is brown. Multiplying this color with an underlying red layer creates a red bag; a green layer creates a green bag; etc.

   Tip: By adjusting the fill color of the Secondary Color layer and the Main Color layer, you can change the image's dark and light colors, respectively. This effect mirrors the one used to generate color variations in Viewpoint Media Player.

   Note: The handbag image in our Photoshop sample illustration cannot have a white color applied to it since the original image (the bag) is dark. To create a white image, lighten and de-saturate the image.

Create a mask alpha channel layer

1. "Control-click" your mask layer by positioning the cursor over the layer on the Layer Palette and then holding down the **Control (Ctrl)** key while clicking the mouse.
2. Select the layer containing your image's highlights.
3. From the **Edit** menu, select the **Fill** option to access the Fill dialog box.
4. From the **Fill** dialog box, click the **Use** menu and select the fill color, white.
5. From the **Select** menu, create a new layer and choose **Deselect**.
6. Repeat steps three and four, only this time selecting the fill color, black.

   Note: The fill should affect only the non-swappable areas.

7. From the **Layer** palette, click the new layer and place it underneath the your mask layer.
8. From the **Layer** menu, select **Merge Down**. In our Photoshop sample illustration, we refer to this layer as the Mask Alpha Channel layer. The resulting image combines the alpha channels of the Mask layer with the Highlight layer into a single black and white layer (black is fully transparent; white is fully opaque).

Creating a flat lightmap

1. Create a new 16X16 pixel image.
2. Fill the pixel image with the color gray, as defined by the values: "red 128", "blue 128", "green 128".

> **Note:** By using the color gray as defined by the RGB values 128, 128, 128, you ensure that no lighting effects are added to the image's surface. Lightmaps use colors brighter than the color gray (as defined by the RGB values, 128, 128, 128) as highlights and colors darker than the color gray (as defined by the RGB values 128, 128, 128) as shadows.

Exporting the Photoshop layers

1. From the **File** menu, export the Original Image layer as a .jpg file. In our Photoshop sample illustration, we name this file, "bag.jpg".

> **Tip:** Before saving your original image as a .jpg file, select the original image layer and sharpen it (go to the **Filter** menu, click the **Sharpen** menu and select Sharpen) to avoid image blurring during the 3D rendering process.

2. Export the Mask layer as a .jpg file. For example, we name this layer, "mask.jpg".

> **Note:** .jpg files do not support transparency so when you export this layer, its background fills with the color white. This white area creates the image's highlights.

3. Export the Mask Alpha Channel layer as a .jpg file. For example, we name this layer, "mask_alpha.jpg".
4. Export the vector shapes on the Secondary Color layer as an Illustrator (.ai) file. For example, we name this layer, "shape.ai".
5. Export the flat lightmap as a .jpg file. For example, we name this lightmap, "flat_light.jpg".

Step 2: Building Swappable Areas in 3ds max

Step two of the color swapping process requires you to build the swappable areas of your product display image. To do this, you can use any 3D authoring tool, such as 3ds max, which is the program we use in our instructions.

Process Overview

The following steps present the high-level 3ds max modeling process for enabling your product display images with color swapping functionality. Each step is explained in more detail below.

1. "Importing your Illustrator (.ai) file into 3ds max" — Importing your file gets you started!
2. "Converting the objects to Editable Mesh" — Editable Meshes convert your scene's objects from outline shapes to polygons, enabling them to be mapped with the product display image.

3 "Adding UVW mapping to your selected objects" — UVW mapping sets the coordinates on which the mapped image is applied.

4 "Preparing your objects for color swapping" — Attaching your objects simplifies your scene by compiling like objects into single objects.

5 "Creating a mask object" — The mask object is the plane on which the mask image is mapped.

6 "Positioning secondary color object(s)" — The secondary color object(s) are positioned between the mask object and the main color object, as illustrated below.

7 "Adding textures to your objects" — The main color object and the secondary color object(s) are mapped with the original image.

8 "Adding a camera to your scene" — The camera makes your product display image viewable.

9 "Saving and exporting your scene" — Exporting the .ase file allows you to begin refining your image in Scene Builder.

Sample 3ds max Illustration

The following graphic illustrates the second step of the ImageLayer color swapping process, broken into three sections, each describing the modeling components of the process.

Step-by-Step Instructions

In our sample color swapping project, we built the swappable areas of the handbag using the following procedures.

Importing your Illustrator (.ai) file into 3ds max

1. Go to the **File** menu; select **Import** and browse to select the 3ds max.
2. From the Select File to Import dialog box, browse to your Photoshop-exported .ai file and click **Open**. For example, we named this file "shape.ai".
3. In the AI Import dialog box, select **Completely Replace this Scene** and then click **OK**.
4. In the Shape Import dialog box, select **Multiple Objects** and then click **OK**.

   Note: When you import your shapes into 3ds max, they are automatically selected.

5. From the left viewport, rotate all of your shapes -90 degrees along the world z-axis to orient the shapes so they are upright in world space.

Converting the objects to Editable Mesh

1. Right-click the selected objects to display a context-sensitive menu. From this menu, choose **Convert To:**.
2. From the **Convert To:** menu, select **Editable Mesh**, thereby converting the objects from empty outlines to filled polygon planes.

Adding UVW mapping to your selected objects

1. From the **Tools** panel, with the objects still selected, click the **Modify** tab to access the **Modifier List**.
2. From the **Modifier List**, select **UVW Map** to add a UVW map to your selected objects.

Preparing your objects for color swapping

1. Select an object in your product display scene and click the **Modify** tab.
2. From the **Modify** tab, expand the **Modifier List** menu.
3. From the **Modifier List** menu, select **Edit Mesh**.

   Important: Skip step 4 if you have only two objects in the scene.

4. From the **Edit Mesh** panel, attach the other objects together-except for the image bounds rectangle-by clicking **Attach List**.

   Note: Attaching the objects combines the primary and secondary swappable areas into one shape.

5. From the **Command** panel, give the attached object a unique name. For example, we name this object, "secondary_color".
6. Give the remaining object, the rectangle, a unique name. For example, we name the rectangle object, "main_color".

Creating a mask object

1. Select the rectangle object. For example, we name this object, "main_color".

   Note: To create a mask, we create a plane above the color shape objects, duplicating the rectangle shape ("main_color") and positioning it in front of the other objects.

2. Press the **Shift** key and, using the **Move** tool, drag the object up along the world z-axis.

   Tip: Position the cloned object just above the original, ensuring that the two objects are separate but close enough to avoid distorted rendering problems.

3. Release the mouse to access the ensuing dialog box.
4. From this dialog box, find the **Object** section and select **Copy**.
5. Give the object a unique name. For example, we name this object, "mask".

Positioning secondary color object(s)

1. If your project includes secondary swappable areas, select the secondary color object. For example, we name this object, "secondary_color".
2. Drag the secondary color object up along the world z-axis with the Move tool and position it between the rectangle object and its cloned object. For example, we position the "secondary_color" object between the "main_color" and "mask" planes.

Adding textures to your objects

1. Add texture to your "main_color" object by:
   - setting the **Diffuse** color to white;
   - specifying the original bag ("bag.jpg") as the Diffuse map; and
   - assigning your flat lightmap .jpg file to the "main_color" object's Reflection map. For example, in our sample project, in Photoshop we name this file, "flat_light.jpg".

   Note: Skip step 2 if your project involves only one swappable area.

2. Add texture to your "secondary_color" object by:
   - setting the **Diffuse** color to white;
   - specifying the original bag ("bag.jpg") as the Diffuse map; and
   - assigning your flat lightmap .jpg file to the "secondary_color" object's Reflection map. For example, in our sample project, in Photoshop we name this file, "flat_light.jpg".
3. Add texture to your "mask" object by:
   - setting the mask image ("mask.jpg") as the Diffuse map; and
   - specifying the mask alpha ("mask_alpha.jpg") as the Opacity map.

Adding a camera to your scene

1. Create a camera with a long focal length (200mm).

   Tip: Use a camera with a long focal length to minimize the perspective distortion in the your product display image.

2. Center the camera and position it at a perpendicular angle to your objects' planes.

   Tip: Ensure that the resulting scene view corresponds to your project's requirements.

Saving and exporting your scene

1. Go to the **File** menu and choose **Save**.
2. Give your scene a unique name.
3. Export the scene as an .ase file. Be sure that the following checkboxes are marked from the Export File dialog box: Materials, Geometric, Shapes, and Cameras.

Step 3: Assembling a Swappable Scene in Scene Builder

Step three of the color swapping process requires you to assemble your product display image's scene, refining details such as cameras and render mode. We use Viewpoint Scene Builder to effect these scene changes.

Process Overview

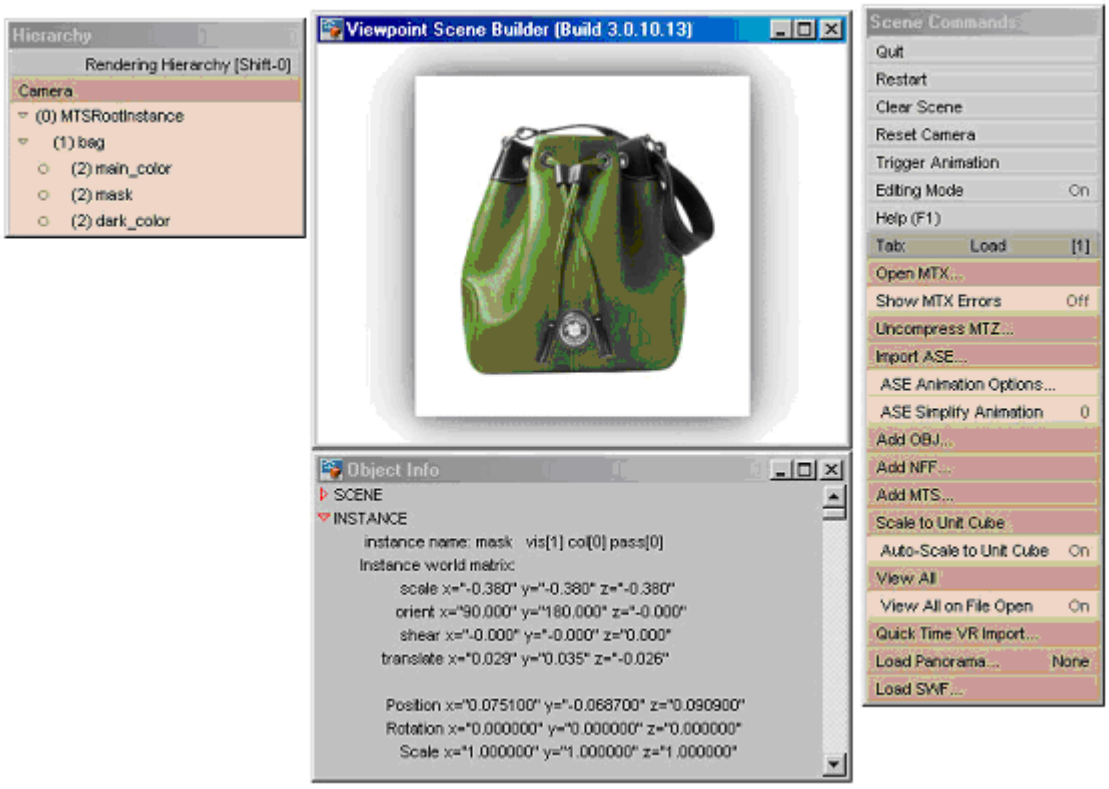
The following steps present the high-level scene refinement and publication process for enabling your product display images with color swapping functionality. Each step is explained in more detail below.

1. "Importing your 3ds max .ase file into Scene Builder" — Now you can start!
2. "Setting your scene's render mode" — The prescribed Lightmap TexMod render mode combines texture and lightmap lighting and multiplies the combination with the geometry's diffuse color. The combination lighting effect generates multiple color variations from a single texture.
3. "Setting your scene's camera" — The prescribed Still camera mode disables user control of the camera, thereby ensuring that your customers are not able to rotate your product display image.
4. "Publishing your scene and save your control file (.mtx)" — Publishing your scene's .mtx file allows you to embark on the final step, hand editing the MTX code.

Sample Scene Builder Illustration

The following graphic illustrates the third step of the ImageLayer color swapping process. In this example, we have imported the 3ds max (.ase) file into Scene Builder.

> **Note:** When you import your .ase file into Scene Builder, your image may not display as shown in this example. How your image displays depends on the Scene Builder version you use. The steps outlined in this section are valid for most Scene Builder versions, regardless of how your image displays in the preview window.



Step-by-Step Instructions

In our sample color swapping project, we assembled the scene's swappable areas using the following procedures.

Importing your 3ds max .ase file into Scene Builder

1. Open Viewpoint Scene Builder.
2. From the **Load** tab (tab 1), click **View All on File Open** to set it to "off". Set the option to "off" to preserve the camera you created in 3ds max.
3. Go to the **Scene Commands** menu and select the **Load** tab (tab 1).
4. From the **Load** tab, click **Import ASE…** .
5. From the Open 3dsmax ASE File dialog box, browse to the .ase file you created in 3ds max and click **Open**.

Setting your scene's render mode

1. From the **Hierarchy** menu in the left navigation bar, select the secondary color and rectangle objects. For example, we name these objects, "main_color" and "secondary_color".
2. Go to the **Scene Commands** menu and select the **Materials** tab (tab 6); click **Render Mode**.
3. From the **Render Mode** menu, choose **Lightmap TexMod**.
4. Select the rectangle (mask) object.
5. From the **Materials** tab (tab 6), click **Render Mode** and select **Texture**. For example, in 3ds max, we cloned this object from the "main_color" object and named it "mask".

Setting your scene's camera

1. From the **Preferences** tab (tab 9), set the scene width and height dimensions to the dimensions of the original image.
2. Press the **Control (Ctrl)** button and drag your mouse to zoom in the camera until the geometry fits against the edges of the scene.
3. Select the **Globals** tab (tab 5) and click **Camera Navigation**.
4. From the **Camera Navigation** menu, set the camera navigation mode to **Still**.

Publishing your scene and save your control file (.mtx)

1. From the **Scene Commands** menu, select the **Publish** tab (tab 0).
2. From the **Publish** menu, set the **Image Quality** to "85." The "85" setting maintains high image quality.
3. From the **Publish** menu, enter the dimensions of your original image in the **HTML Window Width** and **Height** fields.
4. Go to the Publishing Viewpoint Media Files (MTX/MTS) dialog box and click **Publish**.
5. From the dialog box, browse to the folder where you want to save your scene's control (.mtx) file.
6. Name your control (.mtx) file and click **Save**.

   Note: When publishing, Scene Builder creates a very minimal .mtx file. The next two steps allow you to create a more detailed .mtx file that you can hand edit to complete your ImageLayer-enabled product display image.

7. From the **Publish** tab (tab 0), select the **Save MTX** option.
8. From the Saving MTX Viewpoint Media File dialog box, save over the .mtx file you newly created in step 4.

Step 4: Editing your Scene's Control File

Step four of the color swapping process requires you to hand edit your product display image's control (.mtx) file, adding the animations that change the swappable areas' patterns. We use Notepad to access and hand edit the MTX code.

Process Overview

The following processes address the remaining necessary steps before you can publish your product display image to the web.

1. "Adding animations to your MTX control file" — These animations change the diffuse color of your image's geometry by targeting the rectangle object and the multiple secondary color object materials you created in 3ds max.
2. "Adding navigation buttons to your web page's HTML code" — HTML navigation buttons allow the animations to be triggered.

Step-by-Step Instructions

In our sample color swapping project, we edited the scene MTX control file using the following procedures.

Adding animations to your MTX control file

1. Find the **<MTSTimeElem>** tags that load the scene .mts file.
2. From below the <MTSTimeElem> declaration that loads the .mts file, begin adding your color animations as described in the following sample MTX code:

   Note: The animations load your product display image color options and animate between the current user-selected texture and the previously selected texture.

```
<!-- Animations should be named appropriately. -->
<!-- animation should be set to On="0" -->

<MTSTimeElem Type="Keyframe"  Name="color_1"  On="0" >
```

3. Continuing from the MTX code you created in step 2, animate the diffuse color property, difc, for your materials as described in the following sample MTX code:

```
<!--For example, in our sample color swap project (the bag), the
   materials "secondary_color" and "main_color" are targeted. -->

   <Target Name="MTSMaterial.secondary_color"  Property="difc"
   Timeline="difc_secondary" />
   <Target Name="MTSMaterial.main_color"  Property="difc"
   Timeline="difc_main" />
```

4. Continuing from the MTX code you created in step 3, add the time your animation lasts as described in the following sample MTX code:

```
   <!--The animation lasts .5 seconds -->

   <Time>          0   .5        </Time>
```

5 Continuing from the MTX code you created in step 4, add the final lines of animation code as described in the following MTX code:

> **Note:** The RGB vales for the diffuse color are specified by three 0-1 numbers, highlighted in the following code sample, which you determine for your project. You can replace these values with the exact RGB values you created in Photoshop as described in the following two steps.

```
<Timeline Name="difc_secondary"  Type="3D" >*[.62353 .68627
   .62745]</Timeline>
      <Timeline Name="difc_main"  Type="3D" >*[.54902 .92157
   .58824]</Timeline>
</MTSTimeElem>
```

6 Refer to your preview layer set in the Photoshop file to determine the RGB values for the rectangle and secondary color object colors.

7 Copy the RGB values from the Secondary Color layer and the Original Color layer and paste them as the diffuse color values in the place of the corresponding color objects in Scene Builder.

> **Tip:** Save a version of your control (.mtx) file with another name (for example, "bag_red.mtx") to obtain the RGB color values contained in the <MTSMaterial> declaration (in the "0 - 1" format instead of the "0 - 255" format inherent to PhotoShop and Scene Builder).

8 After you copy the diffuse color values from the Photoshop file for the Original Color and Secondary Color layers, open your control (.mtx) file and find the <MTSMaterial> declaration.

9 Replace the existing RGB values for each layer with the values you copied from Photoshop. In our sample project, we changed the following MTX code:

```
<!--The following code sample includes the diffuse color values for
   the Original Color layer-->

<MTSMaterial Name="main_color"  ID="0"  RenderMode="LightTexMod" >
   <MTSTextureMap Type="Diffuse"  Name="bag" />
   <MTSTextureMap Type="Light"  Name="flat_light_ELight" />
   <MTSColor Type="Diffuse"  r="0.96078"  g="0.54118"  b="0.59216"
   />
</MTSMaterial>

<!--The following code sample includes the diffuse color values for
   the Secondary Color layer-->
<MTSMaterial Name="secondary_color"  ID="0"
   RenderMode="LightTexMod" >
   <MTSTextureMap Type="Diffuse"  Name="bag" />
   <MTSTextureMap Type="Light"  Name="flat_light_ELight" />
   <MTSColor Type="Diffuse"  r="0.84706"  g="0.55294"  b="0.55294"
   />
</MTSMaterial>
```

10 Repeat steps 1 through 9 to add an animation with a unique name for each color variation.

Adding navigation buttons to your web page's HTML code

1. Add navigation buttons to the HTML code into which you embed your product display image.

   Tip: Add a set of buttons to the HTML page to allow the animations to be triggered. For example, when clicked, the HTML button triggers the first color animation from our ImageLayer-enabled sample bag. Add a button, described in the following code sample, to trigger each color animation in your scene:

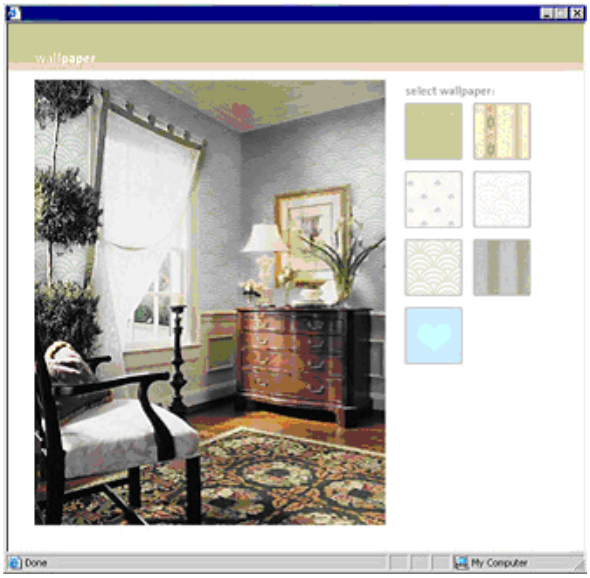
   ```
   <form name="generic">
   <INPUT TYPE=button ID=color_1 VALUE="color_1"
   onClick="vmp.SetProperty('MTSTimeElem.color_1', 'trgr', 1,
   'mts_int')">
   </form>
   ```

2. Now you can view your ImageLayer-enabled product display image in a web browser.

Pattern Swapping

ImageLayer pattern swapping enables your online customers to preview a product in different patterns. You can implement ImageLayer pattern swapping in much the same manner as color swapping, so that when your customers select a new pattern option from the available pattern swatches, the new pattern replaces the existing display pattern. For example, the following illustration provides a typical market application of pattern swapping.

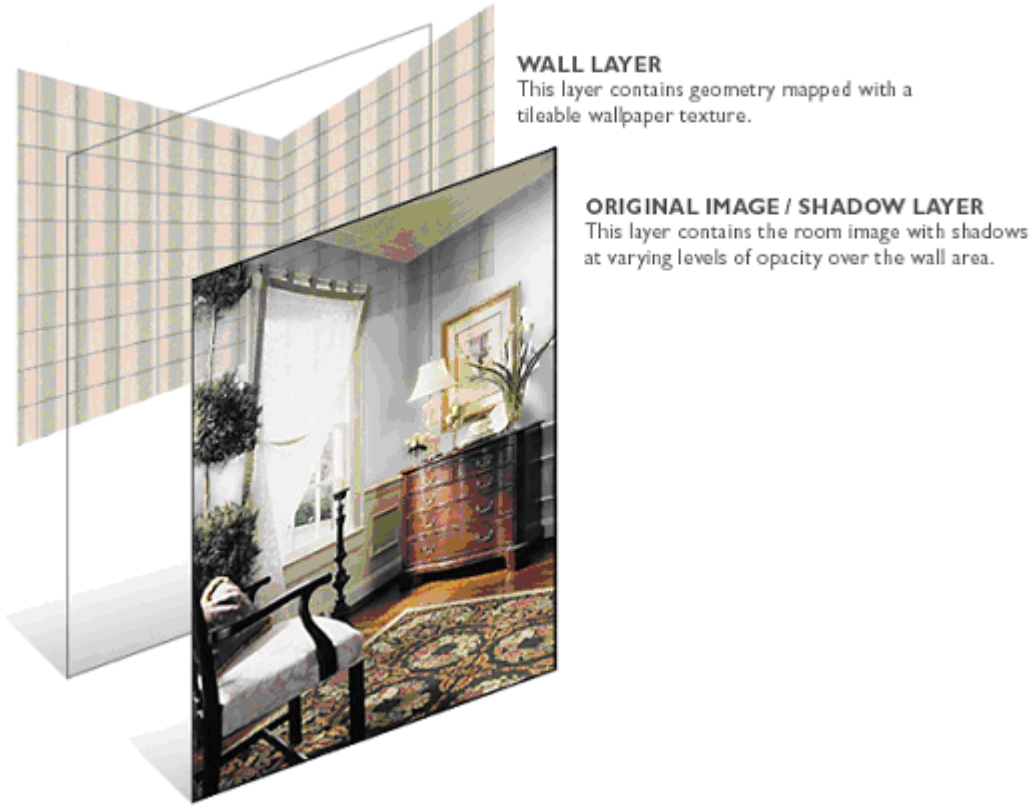
Important: Pattern and fabric swapping becomes more complicated when the pattern or fabric is applied to non-flat surfaces. Keeping a pattern or fabric shape consistent across different surfaces requires complex masking and shadowing. For more information on best practices for choosing a swappable asset, see "Choosing the Right Product Assets for Display".



Note: This product image supports swappable preview patterns that correspond to one pattern area of the wall. It is possible to enable the image with infinite pattern swatch options and multiple pattern areas.

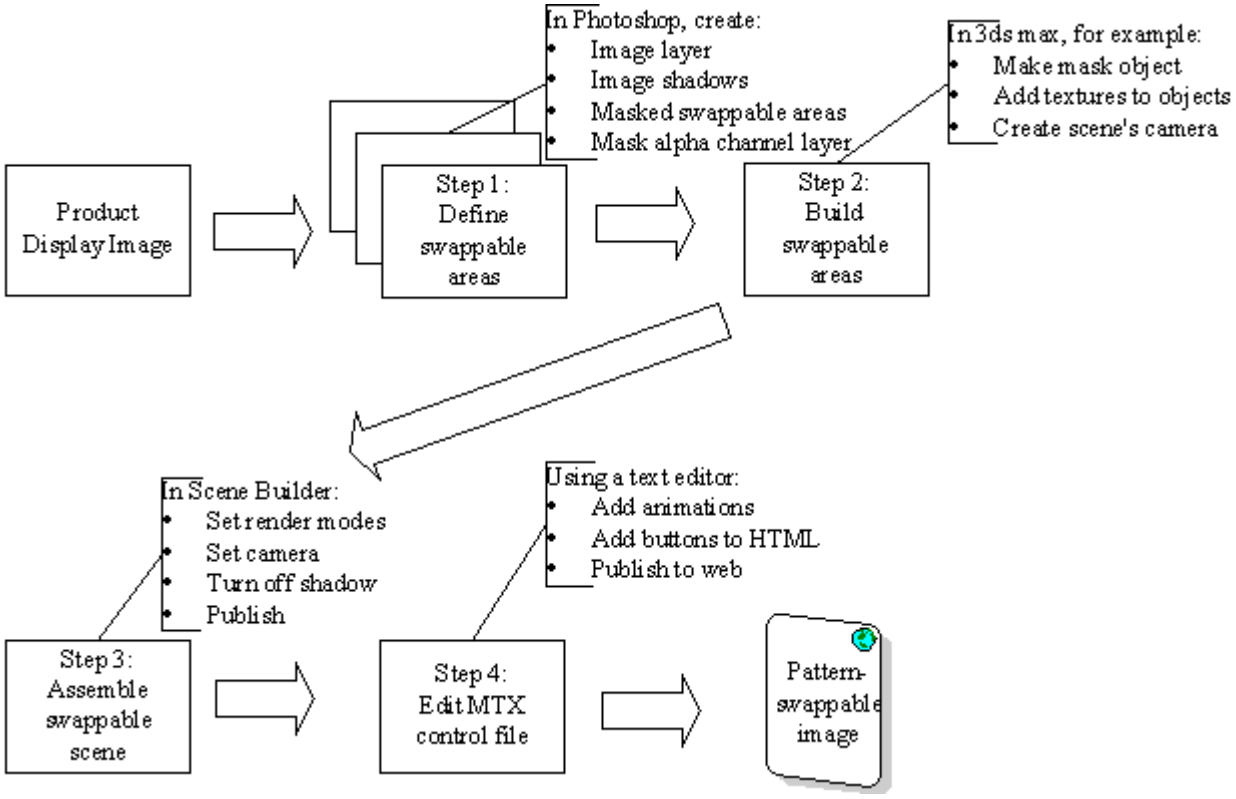
Sample Pattern Swapping Project

The following illustration provides a high-level description of the above pattern swapping project. The image of the room, which contains a pattern-swappable wall, has been separated into two sections, each illustrating a component of the project's requirements.



Workflow

The following graphic illustrates the workflow involved in creating the swappable wall above. Follow these steps to create your own swappable product display images.



Step 1: Defining Swappable Areas in Photoshop

Step one of the pattern swapping process requires you to use Photoshop to mask the swappable areas of your product display image and to define these areas' shadowing.

Process Overview

The following steps present the high-level Photoshop design process for enabling your product display images with pattern swapping functionality. Each step is explained in more detail below.

1. "Opening and saving your image in Photoshop" — Saving your image as a .psd file gets you started!
2. "Extracting shadow from your image" — Adding shadow to your image allows for varying degrees of opacity for the texture swap areas.
3. "Masking out the swappable areas of your image" — Masking defines the areas of your image that your online customers can swap. In the sample image below, these areas include the wall panels.
4. "Creating a mask alpha channel layer" — Creating the mask alpha channel layer enables you to add levels of transparency to your mask layer.

5 "Exporting the Photoshop layers" — Exporting your image's layers allows you to begin the modeling process.

6 "Creating tileable pattern textures" — In a new document, creating the pattern textures you require for your swappable areas provides the your product display image's pattern variations.

Sample Photoshop Illustration

The following illustration exemplifies the first step of the ImageLayer pattern swapping process, broken into two parts (layers), each representing the layers you must create in the process.

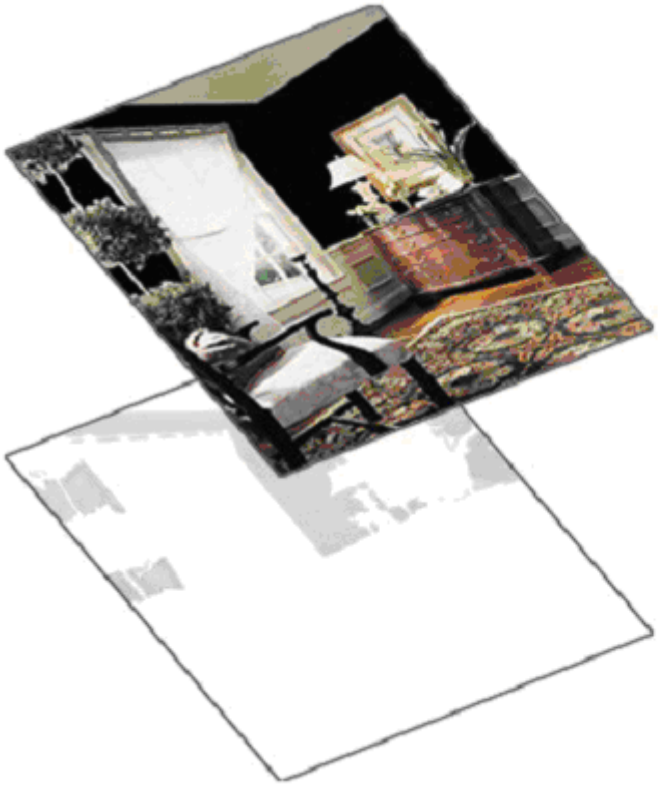


IMAGE LAYER
This layer contains the original image with a black fill for shadows in the areas where textures will be swapped.

ALPHA CHANNEL LAYER
This layer contains the alpha channel to add varying levels of opacity as shadows for the texture swap areas.

Step-by-Step Instructions

In our sample pattern swapping project, we defined the swappable areas of the wall using the following procedures.

Opening and saving your image in Photoshop

1 Open Adobe Photoshop.

2 Go to the **File** menu; click **Open** to browse for and open your product display image.

3 Save your image as a .psd file. For example, this area is described in the Photoshop sample illustration as the Image layer.

Extracting shadow from your image

1. Use the **Pen** tool to trace (mask) the image's swappable areas (the image areas where you want to change pattern).

   Tip: The Pen tool automatically creates a separate layer for your masked shapes. You can use a variety of tools in addition to the Pen tool to mask the nonswappable areas, such as the Magic Wand selection and the Brush tool.

2. Select the alpha channel from your new shadow layer. In our example, we name this layer the Alpha Channel layer.

   Tip: To select the alpha channel, position the cursor over your new shadow layer ("Alpha Channel layer") on the Layer Palette and hold down your keyboard's Control button while clicking the mouse.

3. Select the Image layer.
4. Go to the **Edit** menu and select **Copy**.
5. Select the **Layer** menu and create a new layer to contain the image's shadow areas.
6. Paste the selection into the new layer.
7. Go to the **Image** menu and select **Adjust**.
8. Go to the **Adjust** menu and select **Desaturate** to desaturate the image.
9. From the **Layer** palette, select the **Lock Transparent Pixels** button.

   Note: You lock the transparent pixels on the layer to ensure that the Gaussian Blur, applied in steps 11 and 12, does not erase the edges of the swappable areas.

10. Go to the **Filter** menu, select **Blur**.
11. Go to the **Blur** menu and select **Gaussian Blur**.
12. Use the slider to set a blur value that smoothes out any patterns in the layer.
13. Go to the **Image** menu and select **Adjust** and then **Hue/Saturation**.
14. Adjust the lightness of the layer so that the range is appropriate for the shadow.

    Tip: Hand draw hard edges in your shadowed surfaces to maintain the integrity your surface's lines, depending on how the surfaces' shadowing is affected by the Gaussian Blur value.

Masking out the swappable areas of your image

1. Select your layer's alpha channel.

   Tip: To select the alpha channel, position the cursor over the shadow layer on the Layer Palette and hold down your **Control (Ctrl)** button while clicking the mouse.

2. Select the Image layer.

3 Fill the selection with the color black.

> **Note:** This black area renders at levels of opacity that create the image's shadow.

Creating a mask alpha channel layer

1 Select the layer containing your image's shadows.

2 Go to the **Image** menu and click **Adjustments** and select **Inverse** to invert your image.

Exporting the Photoshop layers

1 From the **File** menu, export the Image layer as a .jpg file. For example, in the context of the Photoshop sample illustration, we name this file, "room.jpg".

2 Export the Image Alpha Channel layer as a .jpg file. For example, we name this layer, "image_alpha.jpg".

Creating tileable pattern textures

1 Open a new document in Photoshop and save it with a unique name.

2 Create a texture of your swappable pattern area that tiles seamlessly.

> **Tip:** Create additional tileable textures as required by your ImageLayer project. You can create your textures from scratch by drawing them or you can use a scanned pattern from a piece of fabric.
>
> **Important:** Since your textures tile across your scene's swappable areas, they do not have to be large in size. All of your tileable textures should conform to 'power of two' size standards (for example, 64X64, 128X128, etc.).

3 Export your tileable pattern textures as a .jpg files.

Step 2: Building Swappable Areas in 3ds max

Step two of the pattern swapping process requires you to build the swappable areas of your product display image. To do this, you can use any 3D authoring tool, such as 3ds max, which is the program we use in our instructions.

Process Overview

The following steps present the high-level 3D authoring process for enabling your product display images with pattern swapping functionality. Each step is explained in more detail below.

1 "Creating a plane for your original image" — This plane must be sized with the same aspect ratio as the original image so the original image can be mapped to it.

2 "Mapping the original image onto the plane" — The alpha channel plane is mapped onto the original image as an opacity map. The alpha channel renders the pattern swappable areas, which are filled with black, at varying levels of opacity to create the image's shadow.

3 "Creating a plane for the pattern swappable areas" — For example, in the sample illustration below, we are enabling the image's walls to be pattern swappable.

4 "Adding a texture map to the pattern swappable areas" — The objective of this step is to set a tileable texture at a frequency that causes the swappable pattern to repeat in the same manner that the wallpaper pattern repeats.

5 "Matching the pattern plane to the perspective of the swappable areas" — By skewing the pattern plane, it can be positioned and shaped to match the perspective surface of the pattern swappable area in the original image. For example, in our sample room scene, the plane is skewed into a bow-tie shape to match the two walls.

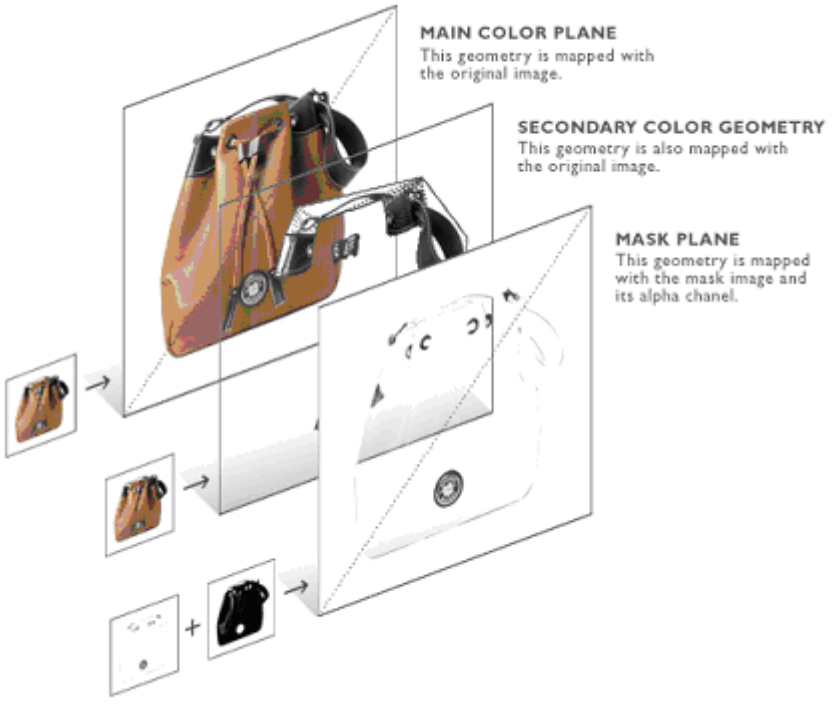
6 "Positioning the swappable area plane behind the image plane" — To create a mask, we create a plane above the pattern swappable objects, duplicating the plane and positioning it in front of the other objects.

7 "Adding a camera to your scene" — The camera makes your product display image viewable.

8 "Saving and exporting your product display image scene" — Exporting the .ase file allows you to begin refining your image in Scene Builder.

Sample 3ds max Illustration

The following illustration exemplifies the second step of the ImageLayer pattern swap process, broken into two sections, each describing the modeling components of the process.



Step-by-Step Instructions

In our sample pattern swapping project, we built the swappable areas of the wall using the following procedures.

Creating a plane for your original image

1. Open 3ds max.
2. From the **Tool** panel, select the **Create** tab.
3. From the front viewport, create a new plane. In our example, we name this plane, "image_plane".

> **Note:** The aspect ratio of the plane should be the same as the aspect ratio of the image. For example, if your image is 400x500 pixels, make your plane's dimensions 400x500 units.

Mapping the original image onto the plane

1. Add UVW mapping to the plane.
2. Add texture to the original image object by:
   - setting the Diffuse color to white;
   - specifying the original image ("room.jpg") as the Diffuse map; and
   - specifying the alpha mask as the Opacity map.
3. Map your image onto the plane. In our sample project, in Photoshop we named this file, "image_alpha.jpg".

Creating a plane for the pattern swappable areas

1. From the **Tool** panel, select the **Create** tab.
2. Create a new plane. For example, we name this plane, "texture_plane".

> **Tip:** Make this plane roughly the size of the swappable area (in our example, the walls).

Adding a texture map to the pattern swappable areas

1. From the **Tool** panel, select the **Modify** tab and click the **Modifier** list.
2. From the **Modifier** list, select **UVW Map** and set a tileable texture at a frequency that causes the swappable pattern to repeat in the same manner that the wallpaper pattern repeats.

> **Tip:** Ensure that the UVW mapping tile's aspect ratio matches the original image's aspect ratio.

Matching the pattern plane to the perspective of the swappable areas

1. From the **Tab** panel, select the **Modify** tab and click the **Modifier** list.
2. From the **Modifier** list, select **FFD** (box) to access the FFD dialog box.

   Tip: Depending on the shape you are matching, you can use different tools to accomplish this step. For the bowtie shape we use in our example, the FFD (Free Form Deformation) tool works best; but other techniques, such as Editable Patches can work as well.
3. From the Dimensions section of the FFD dialog box, click the **Set Number of Points** button and set the dimensions to "3x3x3."
4. Set the **Tension and Continuity** fields to "0."
5. Use the FFD control points to skew the swappable area plane to match the perspective in the image. In our example, the plane was skewed into a bowtie shape to match the perspective of the walls in the image.

Positioning the swappable area plane behind the image plane

1. Select the swappable area plane. For example, we name this plane, "texture_plane".
2. Use the **Move** tool to drag the object up along the world z-axis.

   Tip: Position the object's texture plane just behind the original, ensuring that the two objects are separate but close enough to avoid distorted rendering problems.

Adding a camera to your scene

1. Create a camera with a long focal length (200mm).

   Tip: Use a camera with a long focal length to minimize the perspective distortion in the your product display image.
2. Center the camera and position it at a perpendicular angle to your objects' planes.

   Tip: Ensure that the resulting scene view corresponds to your project's requirements.

Saving and exporting your product display image scene

1. From the **File** menu, choose **Save**.
2. Give your scene a unique name. For example, we name our scene, "room.max".
3. Export the scene as an .ase file.

   Important: Be sure to mark the following checkboxes from the Export File dialog box: Materials, Geometric, Shapes, and Cameras.

Step 3: Assembling a Swappable Scene in Scene Builder

Step three of the pattern swapping process requires you to assemble your product display image's scene, refining details such as cameras and render mode. We use Viewpoint Scene Builder to effect these scene changes.

Process Overview

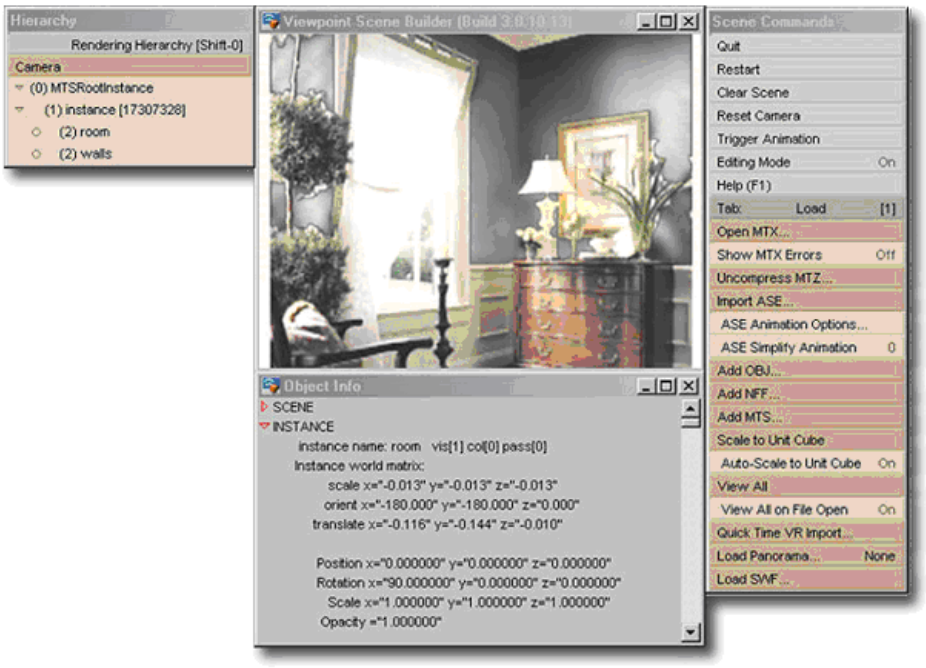
The following steps present the high-level scene assembly process for enabling your product display images with pattern swapping functionality. Each step is explained in more detail below.

1. "Importing your 3ds max .ase file into Scene Builder" — Now you can start!
2. "Setting your scene render mode" — The prescribed Texture render mode does not require a lightmap because it renders the texture without lighting effects.
3. "Setting your scene's camera" — The prescribed Still camera mode disables user control of the camera, thereby ensuring that your customers are not able to rotate your product display image.
4. "Publishing your scene and save your control file (.mtx)" — Publishing your scene's .mtx file allows you to embark on the final step, hand editing the MTX code.

Sample Scene Builder Illustration

The following graphic illustrates the third step of the ImageLayer pattern swap process. In this example, we have imported the 3ds max (.ase) file into Scene Builder to set the scene's render mode and camera to publish the scene.

> **Note:** When you import your .ase file into Scene Builder, your image may not display as shown in this example. How your image displays depends on the Scene Builder version you use. The steps outlined in this section are valid for most Scene Builder versions, regardless of how your image displays in the preview window.



Step-by-Step Instructions

In our sample pattern swapping project, we assembled the scene's swappable areas using the following procedures.

Importing your 3ds max .ase file into Scene Builder

1. Open Viewpoint Scene Builder.
2. Go to the **Scene Commands** menu and select the **Load** tab (tab 1).
3. Click **Import ASE…** .
4. From the Open 3dsmax ASE File dialog box, browse to the .ase file you created in 3ds max and click Open.

Setting your scene render mode

1. From the **Hierarchy** menu in the left navigation bar, select the swappable and nonswappable objects. In our sample project, we name these objects, "room" and "wall".
2. From the **Scene Commands** menu, select the **Textures** tab (tab 6).
3. From the **Textures** menu, click **Render Mode**.
4. From the **Render Mode** menu, select **Texture**.

> **Note:** The Texture render mode does not require a lightmap because it renders the texture without lighting effects.

Setting your scene's camera

1. From the **Preferences** tab (tab 9), set the scene's width and height dimensions to the dimensions of the original image.
2. Press the **Control (Ctrl)** button and drag your mouse to zoom in the camera until the geometry fits against the edges of the scene.
3. Select the **Globals** tab (tab 5) and click **Camera Navigation**.
4. From the **Camera Navigation** menu, set the camera navigation mode to **Still**.

> **Note:** The Still mode disables user control of the camera, thereby ensuring that your customers are not able to rotate your product display image.

Publishing your scene and save your control file (.mtx)

1. From the **Scene Commands** menu, select the **Publish** tab (tab 0).
2. From the **Publish** menu., set the **Image Quality** to "85." The "85" setting maintains high image quality.
3. From the **Publish** menu, enter the dimensions of your original image in the **HTML Window Width and Height** fields.
4. Click the **Publish** option and from the Publishing Viewpoint Media Files (MTX/MTS) dialog box, browse to the folder where you want to save your scene (control file).
5. Give your control (.mtx) file a unique name and click Save.

> **Note:** When publishing, Scene Builder creates a very minimal .mtx file. The final two steps of this process allow you to create a more detailed .mtx file that you can hand edit to complete your ImageLayer-enabled product display image.

6. From the **Hierarchy** menu in the left navigation bar, click your swappable object to expand its menu. In our example, we click on the wall object to access the wall material ("wall_tile3", for example).
7. From the expanded menu, select your swappable object's material.
8. From the **Scene Commands** menu, select the **Textures** tab (tab 6).
9. From the **Textures** menu, select **Texture (Diffuse)** to load your texture .jpg file.

> **Note:** Loading this texture reapplies the object's texture, making the pattern swappable texture an external asset, which is necessary so that the texture can be swapped for another.

10. From the **Publish** tab (tab 0), select **Save MTX**.
11. From the Saving MTX Viewpoint Media File dialog box, save over the .mtx file you created in step 4.

> **Note:** By saving the .mtx file in this manner, Scene Builder creates a more detailed .mtx file from which you can hand edit to complete your ImageLayer-enabled product display image.

Step 4: Editing your Scene's Control File

Step four of the pattern swapping process requires you to hand edit your product display image's control (.mtx) file, adding the animations that change the swappable areas' patterns. We use Notepad to access and hand edit the MTX code.

Process Overview

The following processes address the remaining necessary steps before you can publish your product display image to the web.

1. "Adding animations to your MTX control file" — These animations change the diffuse color of your image's geometry by targeting the materials you created in 3ds max.
2. "Adding user interface navigation to your web page's HTML code" — A set of HTML navigation buttons allows the animations to be triggered.

Step-by-Step Instructions

In our sample pattern swapping project, we edited the scene's MTX control file using the following procedures.

Adding animations to your MTX control file

1. Find the <MTSTimeElem> tags that load your scene's .mts file.
2. From below the <MTSTimeElem> declaration that loads your scene's .mts file, Animate your swappable materials as described in the following code snippet, taken from our sample scene's MTX control file:

> **Note:** The animations load your product display image's texture options and animate between the current user-selected texture and the previously selected texture.

```
<!-- This MTSTimeElem declaration loads the texture according to
       the path
       you specify. The path dynamically updates when the user
       clicks this
       pattern option. -->

   <MTSTimeElem Name="new_loader" Type="MTSImageStream" On="0"
   AutoStop="1"
      Path="path">
      <Target Name="MTSTexture.new_texture"/>
   </MTSTimeElem>

    <!-- This MTSTimeElem declaration creates a fade effect when
          new texture
      is loaded, defined by the <Time> values. -->
   <MTSTimeElem Name="animate_texture" Type="Keyframe" On="0" >
          <Target Name="MTSTexture.wall_texture"  Property="pixl"
             Timeline="T1" />

          <Time>
             0      .5
          </Time>

          <Timeline Name="T1"  Type="Texture" >
          [MTSTexture.wall_texture] [MTSTexture.new_texture]
          </Timeline>
      </MTSTimeElem>
```

Tip: Ensure that the material names you include in your code match the material names in your files.

3 Add an interactor to catch download done events for each new texture, as described in the following code snippet from our sample scene's MTX control file:

```
<!-- This MTSInteractor catches the download done event of the new
       texture and triggers the fade animation -->

<MTSInteractor>
    <MTSHandle Event="MTSLoadDone:new_loader"
       Action="MTSAssignProperty"
       Target="MTSTimeElem.animate_texture::trgr" Value="1"/>
</MTSInteractor>
```

4 Repeat steps 1 through 3 to add an animation with a unique name for each pattern variation.

Adding user interface navigation to your web page's HTML code

1 Add user interface navigation to the HTML code into which you embed your product display image.

Tip: Use preview versions of your pattern textures to show your customers which pattern options they can choose. Or, add a set of buttons (or the user interface method of your choice) to the HTML page. The user interface you choose to build allows the animations to be triggered.

HTML User Interface Example: When clicked, the user interface triggers the first pattern animation from our ImageLayer-enabled sample wall. Add a pattern swatch, described in the following code sample, to trigger each pattern animation in your scene:

```
<A HREF="javascript:swapTexture('images/wall_tile2.jpg')" >
<IMG SRC="images/tile_2.gif" BORDER="0" />
```

JavaScript Triggering Example: For the triggering mechanism to work, add the following JavaScript code:

```
<script language="JavaScript">
// this function passes a path to a new texture to Viewpoint
Media Player
// and triggers the animation that loads the new texture
function swapTexture(path) {
vmp.SetProperty('new_loader', 'Path', path, 'mts_str');
vmp.TriggerAnim('MTSTimeElem.new_loader');
}
</script>
```

2 Now you can view your ImageLayer product display image in a web browser.

Appendix A: Help, Resources, and Feedback

Viewpoint Developer Central: A Complete Resource

Viewpoint Developer Central is a complete resource for Viewpoint content developers. From this website, you can obtain Viewpoint applications, user guides, downloadable example files, support, production tips, and techniques – to name just a few.

Use Viewpoint Developer Central to:

- Get Assistance — For questions about using Viewpoint Technology, click **Support** in the left navigation bar and then select **Forums**.
- Get Examples — Click **Examples & Tips** in the left navigation bar.
- Subscribe to the Viewpoint Developer Newsletter — Learn new production tips and techniques for creating 3D and rich media content for the web with Viewpoint Technology. Click **Newsletter** in the left navigation bar.
- Give Feedback About Viewpoint Applications — Viewpoint Corporation values your feedback. Direct your comments and suggestions to the **Viewpoint Forums**.

You can also visit the Viewpoint Corporation website for company news, links to websites featuring Viewpoint Technology, and more.

Download Viewpoint Applications, Guides, and Examples

Viewpoint Developer Central is continuously updated with the latest versions of applications, user guides, and examples. Links to these features are provided in the left navigation bar.

Viewpoint Applications

You can download Viewpoint applications free of charge. Applications available for download include:

- Viewpoint Media Player — The web browser plug-in necessary to view Viewpoint content with Netscape Navigator or Internet Explorer.
- Viewpoint Scene Builder — An essential application for assembling a scene and publishing it in .html/.mtx/.mts format.
- Viewpoint Media Publisher — An application enabling you to quickly create Viewpoint Technology web applications from Viewpoint media files (.mtx/.mtz) by embedding them in web (.html) pages or running transformations on .mtx (XML) files through built-in XSLT support.
- Viewpoint Stream Tuning Studio — An application for reducing .mts file sizes, enabling optimized 3D scenes rendered on a web page to stream quickly and retain visual integrity.
- Viewpoint Control Panel — A utility for checking, installing, and removing individual Viewpoint Media Player components.

User Guides

Access Viewpoint documentation from the **Documentation** link on Viewpoint Developer Central.

Appendix B: Sample MTX Control Files

To view a complete ImageLayer scene, refer to the following two sample .mtx files for color swapping and pattern swapping product display images.

Sample Color Swapping .mtx File

This sample .mtx file includes elements of the following illustration, such as the material textures (declared between the <MTSMaterial> open and close tags) for both the bag's main and secondary color swapping areas and the actual color swapping animations (declared between the <MTSTimeElem Type="Keyframe"> open and close tags).

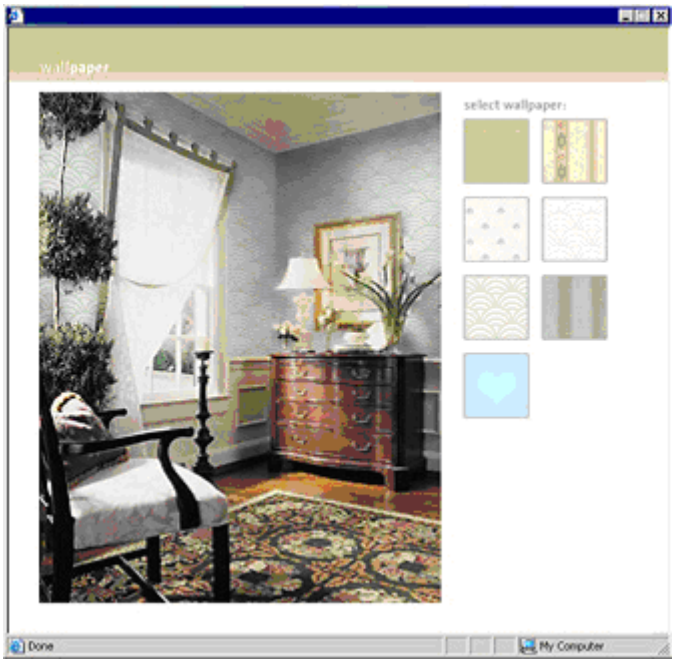
```
<?xml version="1.0"?>
<!-- Viewpoint Experience Technology scene description file. -->
<!-- Created: Wed Jan 15 15:50:53 2003 -->
<!-- Creator: Viewpoint Scene Builder (Build 3.0.11.131) -->
<!-- Author: rcase -->

<MTSScene Version="311" >
    <MTSSceneParms RenderMode="LightTexture"  BlendShadow="1"
    EdgeBias="1" >
        <Transform>
            <Scale x="0.37995"  y="0.37995"  z="0.37995" />
        </Transform>
    </MTSSceneParms>

    <MTSCamera Mode="Still"  CameraUnitScale="0.379946 0.379946
    0.379946"
    OrbitDist="5.07135" >
        <Rotate x="90"  y="0"  z="-0.63029" />
        <Scale x="0.99996"  y="1"  z="0.2" />
        <ViewLocation x="-0.34214"  y="-25.35442"  z="0.0329" />
        <LookAt x="90"  y="0"  z="-0.63029" />
    </MTSCamera>

    <MTSInstance Name="bag" >
        <Transform>
            <Scale x="-1"  y="-1"  z="-1" />
            <Shear xy="0"  yz="0"  xz="0" />
            <Rotate x="90.00003"  y="-180"  z="-0.00003" />
        </Transform>

        <MTSInstance Name="main_color" >
            <Transform>
                <Scale/>
                <Position x="0.0751"  y="-0.0687"  z="0" />
            </Transform>

            <MTSGeometry Name="main_color_GEOM"  MultiUV="0"
            BackFaceDir="-1" />
            <!--Material textures -->
```

```
        <MTSMaterial Name="main_color"  ID="0"
        RenderMode="LightTexMod" >
            <MTSTextureMap Type="Diffuse"  Name="bag"  x="512"
            y="512" />
            <MTSTextureMap Type="Light"  Name="flat_light_ELight"
            x="256"
            y="256" />
            <MTSColor Type="Diffuse"  r="1"  g="1"  b="1" />
        </MTSMaterial>
    </MTSInstance>

    <MTSInstance Name="mask" >
        <Transform>
            <Scale/>
            <Position x="0.0751"  y="-0.0687"  z="0.0909" />
        </Transform>

        <MTSGeometry Name="mask_GEOM"  NoAAOpenEdges="1"
        MultiUV="0"
        BackFaceDir="-1" />

        <MTSMaterial Name="mask"  ID="0"  RenderMode="Texture" >
            <MTSTextureMap Type="Diffuse"  Name="mask"  HasAlpha="1"
            x="512"
            y="512" />
            <MTSTextureMap Type="Light"  Name="flat_light_ELight"
            x="256"
            y="256" />
        </MTSMaterial>
    </MTSInstance>

    <MTSInstance Name="secondary_color" >
        <Transform>
            <Scale/>
            <Position x="0.3963"  y="-1.4628"  z="0.0461" />
        </Transform>

        <MTSGeometry Name="secondary_color_GEOM"  NoAAMaterials="1"
        NoAAOpenEdges="1"  MultiUV="0"  BackFaceDir="-1" />

        <MTSMaterial Name="secondary_color"  ID="0"
        RenderMode="LightTexMod">
            <MTSTextureMap Type="Diffuse"  Name="bag"  x="512"
            y="512" />
            <MTSTextureMap Type="Light"  Name="flat_light_ELight"
            x="256"
            y="256" />
            <MTSColor Type="Diffuse"  r="1"  g="1"  b="1" />
        </MTSMaterial>
    </MTSInstance>
</MTSInstance>

<MTSTimeElem Type="MTSStream"  Name="bag"  Path="bag.mts" >
    <Target Name="MTSInstance.bag" />
</MTSTimeElem>

<MTSTimeElem Type="MTSImageStream"  On="1"  Path="flat_light.jpg" >
    <Target Name="MTSTexture.flat_light_ELight" />
</MTSTimeElem>

 <!-- First color animation -->
 <MTSTimeElem Type="Keyframe"  Name="color_1"  On="0" >
    <Target Name="MTSMaterial.secondary_color"  Property="difc"
    Timeline="difc_secondary" />

    <Target Name="MTSMaterial.main_color"  Property="difc"
    Timeline="difc_main" />

    <Time>           0   .5        </Time>
```

```
      <Timeline Name="difc_secondary"  Type="3D" >*[.62353 .68627
      .62745]   </Timeline>

      <Timeline Name="difc_main"  Type="3D" >*[.54902 .92157 .58824]
</Timeline>
   </MTSTimeElem>

    <!-- Second color animation -->
    <MTSTimeElem Type="Keyframe"  Name="color_2"  On="0" >
      <Target Name="MTSMaterial.secondary_color"  Property="difc"
      Timeline="difc_secondary" />

      <Target Name="MTSMaterial.main_color"  Property="difc"
      Timeline="difc_main" />

      <Time>          0   .5        </Time>

      <Timeline Name="difc_secondary"  Type="3D" >*[.84706 .55294
      .55294]   </Timeline>

      <Timeline Name="difc_main"  Type="3D" >*[.31373 .4 .62745]
</Timeline>
   </MTSTimeElem>

    <!-- Third color animation -->
    <MTSTimeElem Type="Keyframe"  Name="color_3"  On="0" >
      <Target Name="MTSMaterial.secondary_color"  Property="difc"
      Timeline="difc_secondary" />

      <Target Name="MTSMaterial.main_color"  Property="difc"
      Timeline="difc_main" />

      <Time>          0   .5        </Time>

      <Timeline Name="difc_secondary"  Type="3D" >*[.84706 .55294
      .55294]   </Timeline>

      <Timeline Name="difc_main"  Type="3D" >*[.96078 .54118 .59216]
</Timeline>
   </MTSTimeElem>

    <!-- Fourth color animation -->
    <MTSTimeElem Type="Keyframe"  Name="color_4"  On="0" >
      <Target Name="MTSMaterial.secondary_color"  Property="difc"
      Timeline="difc_secondary" />

      <Target Name="MTSMaterial.main_color"  Property="difc"
      Timeline="difc_main" />

      <Time>          0   .5        </Time>

      <Timeline Name="difc_secondary"  Type="3D" >*[1 1 1]</Timeline>
      <Timeline Name="difc_main"  Type="3D" >*[1 1 1]</Timeline>
   </MTSTimeElem>

</MTSScene>
```

Sample Pattern Swapping .mtx File

This sample .mtx file includes elements of the following illustration, such as the pattern swappable texture animations (declared after the <MTSTimeElem> open and close tags that load the .mts file).




```
<?xml version="1.0"?>
<!-- Viewpoint Experience Technology scene description file. -->
<!-- Created: Wed Jul 17 10:21:25 2002 -->
<!-- Creator: Viewpoint Scene Builder (Build 3.0.10.70) -->
<!-- Author: rcase -->

<MTSScene Version="308" >
<!--
<MTSSceneParms RenderMode="LightTexture"  BlendShadow="1"
EdgeBias="1" >
       <Transform>
          <Scale x="0.00299"  y="0.00299"  z="0.00299" />
          <Position x="-0.10326"  y="-0.55853"  z="0" />
       </Transform>
   </MTSSceneParms>

   <MTSCamera Mode="Still"  >
      <ViewLocation x="-0.102"  y="0.418"  z="1.177" />
   </MTSCamera>

    <MTSSceneParms RenderMode="LightTexture"  BlendShadow="1"
      EdgeBias="1" >
      <Transform>
          <Scale x="0.00299"  y="0.00299"  z="0.00299" />
          <Position x="-0.10326"  y="-0.55853"  z="0" />
      </Transform>
   </MTSSceneParms>

   <MTSCamera Mode="Still"  OrbitDist="1.17" >
      <ViewLocation x="0.099"  y="0.544"  z="1.17" />
   </MTSCamera>

   <MTSTextureMap Name="room_texture"  HasAlpha="0" />
   <MTSTextureMap Name="room_alpha_texture"  />
```

```
<MTSTextureMap Name="shadow_texture"  />

<MTSTextureMap Name="wall_texture"  />

<MTSTextureMap Name="new_texture" />

 <MTSInstance Name="room"  Collapsed="0">
    <MTSInstance Name="room_child">
        <LayerData Name="room_layer"  Texture="room_texture"
           AlwaysVisible="1"  SrcPin="0 0"  DstPin="0 0"
           AnchorWidget="1"
           WidgetLine="0"  Shadow="0"  WidgetClamp="0" />
    </MTSInstance>
</MTSInstance>


<MTSInstance Name="shadow"  Collapsed="0">
    <MTSInstance Name="shadow_child">
        <LayerData Name="shadow_layer"  Texture="shadow_texture"
              AlwaysVisible="1"  SrcPin="0 0"  DstPin="0 0"
              AnchorWidget="1"
              WidgetLine="0"  Shadow="0"  WidgetClamp="0" />
    </MTSInstance>
</MTSInstance>

 <MTSInstance Name="walls" >
    <Transform>
        <Scale x="-1"  y="-1"  z="-1" />
        <Rotate x="90"  y="-180"  z="0" />
    </Transform>
    <MTSInstance Name="walls" >
        <Transform>
            <Scale x="-1"  y="-1"  z="-1" />
            <Rotate x="-90"  y="0"  z="0" />
            <Position x="201.3342"  y="0"  z="122.5075" />
        </Transform>
        <MTSGeometry Name="walls_GEOM"  BackFaceDir="-1" />
        <MTSMaterial Name="tile_mat"  ID="0"  RenderMode="Texture" >
            <MTSTextureMap Type="Diffuse"  Name="wall_texture" />
        </MTSMaterial>
    </MTSInstance>
</MTSInstance>

 <MTSTimeElem Type="MTSImageStream"  Name="room_texture_loader"
        Path="images/room.jpg"  >
    <Target Name="MTSTexture.room_texture" />
</MTSTimeElem>

<MTSTimeElem Type="MTSImageStream"
    Name="room_texture_alpha_loader"
        UseAsAlpha="1" Path="images/room_alpha.jpg"  >
    <Target Name="MTSTexture.room_texture" />
</MTSTimeElem>

<MTSTimeElem Type="MTSImageStream"  Name="shadow_loader"
    UseAsAlpha="1"
        Path="images/shadow.jpg"  >
    <Target Name="MTSTexture.shadow_texture" />
</MTSTimeElem>

<MTSTimeElem Type="MTSImageStream"  Name="tile_loader" On="1"
        Path="images/wall_tile3.jpg"  >
    <Target Name="MTSTexture.wall_texture" />
</MTSTimeElem>

<MTSTimeElem Type="MTSStream"  Name="walls_loader"
Path="walls.mts" >
    <Target Name="MTSInstance.walls" />
</MTSTimeElem>
```

```
<!--*** This is the ImageStream we pass the path information into ***-
->
    <MTSTimeElem Name="new_loader" Type="MTSImageStream" On="0"
    AutoStop="1"
           Path="blah">
       <Target Name="MTSTexture.new_texture"/>
    </MTSTimeElem>


<!-- *** The texture animator with fade *** -->
    <MTSTimeElem Name="animate_texture" Type="Keyframe" On="0" >
           <Target Name="MTSTexture.wall_texture"  Property="pixl"
              Timeline="T1" />

           <Time>
              0      .5
           </Time>

           <Timeline Name="T1"  Type="Texture" >
           [MTSTexture.wall_texture] [MTSTexture.new_texture]
           </Timeline>
       </MTSTimeElem>


<!--*** This Interactor below triggers the texture animator when the
ImageStream is done ***-->
    <MTSInteractor>
       <MTSHandle Event="AnimationDone:new_loader" Action="Trigger"
          Target="animate_texture" />
    </MTSInteractor>
</MTSScene>
```

Appendix C: ImageLayer XML Elements

This section describes the Viewpoint XML tags, which you should become familiar with to work more effectively with ImageLayer authoring techniques.

ImageLayer Animation Tags and Properties

Animations modify scene elements over time and are contained in an .mtx file in an MTSTimeElem element. Animations are commonly used to modify scene elements (as in a texture animation) and to load additional files into a scene. They can also trigger scripted actions over time (ActionAnimator) and trigger actions from the intersection of geometries (VolumeTrigger).

You must add keyframe texture animations to your ImageLayer control file (.mtx) to enable the smooth transition of one color to the next. All of your scene's texture animations must be declared within an <MTSTimeElem Type=Keyframe"> expression.

<MTSTimeElem Type="Keyframe">

Description	The most commonly used animation, a keyframe is a set of parameters defining a value or a set of values in a transition. For example, a keyframe may define a picture size, a position, or a rotation.

Sample XML Code	**<MTSTimeElem Name="rot_move" Type="Keyframe" On="0">**
	`<Target Name="MTSInstance.filename_MESH_1" Property="rot_" Timeline = "T1"/>`
	`<Time>01 1.52 </Time>`
	`<Timeline Name="T1" Type="3D"> *[ 0 180 0 ][ 0 360 0 ][ 0 90 0 ]`
	`</Timeline>`
	</MTSTimeElem>

Remarks	As of Viewpoint Media Player release 3.0.11, animations that contain only one (1) keyframe set to <Time> 0 </Time> do not work. To fix these animations, either add another keyframe or change the time to something other than "0", such as "0.0001".

<MTSTimeElem> <Target Name>

Description	Specifies the scene entity upon which the animator acts.
Value(s)	A developer-defined name of a scene entity.

Sample XML Code	`<MTSTimeElem Type="Keyframe" On="0">`
	`<Target Name="MTSMaterial.secondary_color"` `Property="difc"`
	`Timeline="difc_secondary" />`
	`</MTSTimeElem>`

Remarks	To ensure that the animator targets the correct scene entity, use a fully defined entity name. This name consists of a prefix that identifies the category of a scene entity that you want to select (namely: MTSInstance, MTSCamera, MTSTexture, MTSMaterial, or MTSTimeElem), followed by a period, and then the scene entity's name. For example, to target an animator to an object that you had named "Sphere", use the fully defined name, "MTSInstance.Sphere".

<MTSTimeElem> <Target Property>

Description	Specifies the property of the entity upon which the animator acts.
Value(s)	`The target's Viewpoint-defined property.`

Sample XML Code	`<MTSTimeElem Type="Keyframe" On="0">`
	`<Target Name="MTSMaterial.secondary_color"`
	`Property="difc"`
	`Timeline="difc_secondary" />`
	`</MTSTimeElem>`

Remarks	In a keyframe animation, this property value defines the type of animation you want to use. For instance, if you use the value "difc", you are setting a texture animation. If you use the value "scl_", you will have a scale animation.

<MTSTimeElem> <Target Timeline>

Description	Specifies the name of the timeline used by the animator.
Value(s)	Developer-defined. The timeline name.

Sample XML Code	`<MTSTimeElem Type="Keyframe" On="0">`
	`  <Target Name="MTSMaterial.secondary_color"`
	`Property="difc"`
	`  **Timeline="difc_secondary" />**`
	`</MTSTimeElem>`

Remarks	If there is only one timeline nested in an MTSTimeElem tag, then you don't have to specify a name for Timeline. **Tip:** Do not use spaces (empty characters) in the name.

Texture Declaration Related Property: difc

Description	Specifies the diffuse color of a texture in RGB (red, green, and blue).
Value(s)	.65, .65, .6.5 Range: 0.0 to 1.0 for each value of RGB.

Sample XML Code	`<MTSTimeElem Type="Keyframe" On="0">`
	`  <Target Name="MTSMaterial.secondary_color"`
	`**Property="difc"**`
	`  Timeline="difc_secondary" />`
	`</MTSTimeElem>`

Remarks	The property difc overwrites a diffuse texture.

Texture-Related Tags and Properties

<MTSInstance> <MTSMaterial>

Description	MTSMaterial is a complex subelement with attributes and nested elements. It contains settings for the color, texture, and material of the parent instance.

Sample XML Code	`**<MTSMaterial Name="main_color"**  ID="0"`
	`RenderMode="LightTexMod" >`
	`  <MTSTextureMap Type="Diffuse"  Name="bag" />`
	`  <MTSTextureMap Type="Light"`
	`Name="flat_light_ELight" />`
	`  <MTSColor Type="Diffuse"  r="0.96078"`
	`g="0.54118"  b="0.59216" />`
	`**</MTSMaterial>**`

Remarks	If you are planning to do any interaction and/or animation with a material, you must name the interaction and/or animation.

<MTSInstance> <MTSMaterial> <MTSColor>

Description	MTSColor is nested within the MTSMaterial tag and its attributes, Type, r, g, and b, determine the material color of a geometry.

Sample XML Code

```
<MTSMaterial Name="main_color"  ID="0"
RenderMode="LightTexMod" >
      <MTSTextureMap Type="Diffuse"  Name="bag" />
      <MTSTextureMap Type="Light"
Name="flat_light_ELight" />
      <MTSColor Type="Diffuse"  r="0.96078"
g="0.54118"  b="0.59216" />
</MTSMaterial>
```

Remarks

None.

Glossary

3D (three-dimensional)	An object or volume that exists in the dimensions of width, height, and depth.
AccumMax	The process of performing multiple anti-aliasing passes while the scene is at rest to refine the image quality. Typically, from 2 to 32 passes achieve a quality comparable to photo-realistic.
action	The transition code (the action) is executed if the start state of the transition matches the current state maintained by the interactor and the current event is equal to the transition event.
.aer	The file extension used by Adobe Atmosphere. It refers to the 3D world that is created with the Atmosphere Builder tool and includes standard objects such as walls, stairs, and other environmental elements.
alpha channel	An optional channel in the texture file that usually defines the transparency of the texture's pixels.
ambient	Light that exists everywhere without a particular source. Ambient light does not cast shadows, but fills in the shadowed areas of a scene.
anchor	Locks a widget to a specific point as defined by the SrcPin (source pin) and DstPin (destination pin) properties.
animation	A motion or transition added to a media atom or a group of media atoms over time. Examples include an object moving around a scene, transitions from one color or texture to another, or an object becoming visible.
anti-aliasing	Anti-aliasing improves the appearance of rendered objects by removing jagged, stair-step edges and averaging colors to create intermediate colors (or shades of gray) in the pixels between contrasting colored regions. Viewpoint Technology provides real-time anti-aliasing while the user interacts with the scene. See also *AccumMax*.
ASCII	American Standard Code for Information Interchange (ASCII). The basis of character sets used in almost all present-day computers.
.ase	ASCII Scene Export file format. Viewpoint Scene Builder imports .ase files from 3ds max (formerly 3D Studio MAX) version 3.1 or greater.
Atmosphere	Refers to Adobe Atmosphere technology in which a 3D world is created that the user can navigate with an avatar.
atoms	See *media atoms*.
avatar	A character used in a 3D world to represent the person navigating the world.
backface culling	The process of removing the unseen polygons that face away from the viewer. This can dramatically speed up the rendering of a polygonal scene.
background rendering	Showing a media atom as the backdrop in the scene. Any objects in the scene appear in front of the background. This is can be done with a panorama, Flash™ movie, or Viewpoint ZoomView.

bilerp	Applies procedural blurring to an image.
billboard	The effect in which an instance (texture or object) always maintains its alignment to the viewer or always faces the camera. The specified side of the instance displays even when the camera is moved around the scene.
bounding box	A rectangular 3D object that represents the maximum extents of an object.
blit	To copy a large array of bits from one part of a computer's memory to the screen.
Broadcast Key	A unique alphanumeric string issued by Viewpoint Corporation to companies or individuals licensed to broadcast Viewpoint Technology content. The string is stored in an .mtx file that is referenced by Viewpoint Technology-enabled web pages. Viewpoint Technology content without a Broadcast Key displays with a watermark. Prior to Viewpoint Media Player version 3.0.8, Broadcast Keys were stored in .txt files.
bump mapping	A method of displaying textures not as a smooth surface, but as a rough surface that responds to different angles of illumination.
camera	The view from which a scene is rendered.
camera constraints	Values added to the camera properties to limit zoom and rotation. This can prevent the user from seeing areas of the object or scene that the content creator wants to hide or can prevent the user from inadvertently moving the object out of the viewable area.
clamp (animation)	Cycles an animation based on the number specified. For instance, Clamp="0" cycles an animation infinitely.
clamp (widget)	Prevents a widget from moving outside the Viewpoint Media Player window.
clipping	Occurs when a media atom is too close to the camera. Polygons closer to the camera than the clipping plane are not rendered.
clipping plane	A virtual plane in which all geometry on one side of the plane is not visible.
collage map	The process of combining several textures into a single texture while preserving the UV coordinates for correct placement of the textures. This can reduce texture sizes and file size.
compositing	The process of combining multiple images or layers into a single image.
crease angle	The angle between two edges.
diffuse	To project light over an entire area of an object; scattered light.
edge	Where two adjacent polygons connect.
edge bias	Removes edge artifacts that may appear when geometry is viewed at close proximity. It is most useful for scenes originally created in third-party software. Adjusting edge bias does not improve scenes created with procedural geometry only.
element	The complete statement of an XML command contained between an opening and closing tag. Elements include attributes and values and may contain nested elements, also known as subelements.

EndState	An optional attribute that defines the default end state of the transition. If during its execution, the transition code does not change the value of the EndState parameter, the current state of the interactor is set to the value specified in the EndState attribute. The value of the state variable of the interactor stays unchanged if the EndState is not specified and the transition code does not change it explicitly.
Event	A special XML tag set defined for user interaction with the mouse. Example: MouseLeftDown. Actions are called by events (also described as messages).
flip polygons	Inverts the geometry of a 3D model.
foreground rendering	Showing any media atom that is not rendered as part of the background.
goemetry	Defines all polygons making up an object.
global	Describes properties added to an entire scene.
hierarchy	A list of things that are linked together in a certain order. A family tree analogy helps describe the parts of the hierarchy. The hierarchy tree traditionally hangs upside down making it easier to read and to use. At the top of the tree is the root, and all things are attached to it. Every object in the hierarchy is called a node. The connections between the nodes are called links. Parent and child nodes are linked in such a way that a parent can have multiple children, but, in this tree, each child can have only one parent. The nodes that have no children are called leaves, because they're at the ends of the tree. Each node can trace its lineage up through the tree to the root. If you choose any parent, then you can call its collection of children a branch of the tree.
instancing	Creating a copy of a specific object in a scene by referencing it. Creating an instance of an object that already exists can help maintain small file size, since the polygons and textures need only be defined once. For example, the tires on a car model can be created this way: The first tire's mesh and texture is defined and the other 3 tires are instanced.
interactors	Elements that allow the user to alter or interact with the scene by clicking or rolling over certain areas within the scene. The programmer defines the interactors in the XML code. OnClick and OnEnter are examples of interactor events.
iPIX	A spherical panorama file format created by Interactive Pictures Corporation that allows users to view images that immerse them in a multi-dimensional, 360° environment.
lightmap	An image that determines how light interacts with and scatters on the surface of an object. Material properties such as diffusion, specularity and reflection are captured in the lightmap. The lightmap in any Viewpoint scene is what the camera sees in any reflective materials of an object. Any spherical image can be used as a lightmap image.
local	Describes properties added to a specific object or set of objects in a scene.
loop	Returning to and playing from the start of a Flash™ movie or animation that has played to its end.

map	To apply a 2D image onto the surface of an object.
materials	Surfaces added to the mesh to give it a finished appearance unlike wireframe rendering.
media atoms	Components of a Viewpoint scene: 3D objects, material properties, sound, object movies, animators, interactors, and the definition of the 3D environment (i.e., panoramas or the maps of environmental lightmaps).
mesh	See *geometry.*
MIP mapping	A technique used to speed rendering of distant textures and remove speckle from distant textures. Multiple textures are generated from a single texture to aid in displaying that texture in a smaller form.
morphing mesh	Describes the effect of altering polygon sizes to cause two objects to appear to meld around one another. For instance, the polygon areas around a character's shoulder and upper arm might morph to create the effect of the objects being connected. Otherwise, the triangles separate and the objects appear to have gaps between them.
MTL	The extension for a text-based file that describes the material associated with an .obj file.
MTS	A binary resource file containing all geometry, materials, and texture information for a Viewpoint Technology scene. MTS is an open-specification 3D file format developed by MetaCreations and Intel Architecture Labs. A key Viewpoint Technology feature is scalability, which automatically maximizes performance and resolution of content according to the available processing power of the user's PC. As a result, web designers can set the detail (frame rate) of these 3D objects to ensure consistent, high-quality viewing. Viewpoint Technology also progressively streams content, allowing users to experience 3D objects from the moment downloading commences, similar to streamed audio or video files. A published Viewpoint Technology scene consists of an .mts file and an .mtx file.
MTX	A Viewpoint XML scene file that contains the hierarchical relationships between objects and other elements in the scene. This file is the script for staging the scene elements and usually references an .mts file.
MTZ	A compressed form of an .mtx file and the preferred format for web-enabled Viewpoint content. Complex animations in an .mtx file can make file size large. Compressing these large .mtx files enables fast downloading of Viewpoint scenes.
MZV	A file format for compressed image tiles (sections of a high-resolution image) used by Viewpoint ZoomView technology.
NFF	Neutral File Format. Files that use a minimal scene-description language in order to test various rendering algorithms and efficiency schemes and to describe the 3D geometry and basic surface characteristics of objects, the placement of lights, and the viewing fulcrum for the eye. Some additional information is provided for aesthetic reasons (such as the color of the objects- not strictly necessary for testing the efficiency of rendering algorithms). Can be imported into Scene Builder.

normal	A vector pointing straight out from the polygon at a right angle that defines the outside surface of the polygon.
normalize	Scales the root instance of the scene.
NURBS	Non-Uniform Rational B-Spline, a mathematical representation of a 3D object. Most CAD/CAM applications support NURBS, which can be used to represent analytic shapes, such as cones, as well as free-form shapes, such as car bodies. NURBS define the curve a surface follows. In contrast, polygons are made up of triangles that define the shape of the surface.
OBJ	Wavefront Object file. A 3D file format that defines the geometry and other properties for objects in Wavefront's Advanced Visualizer. OBJ can also be used to transfer geometric data back and forth between the Advanced Visualizer and other 3D applications. Can be imported into Scene Builder.
panorama	A 360-degree image in which the camera is in the center. QTVR and iPIX panoramas can be used in a Viewpoint scene.
PassClick	In a Viewpoint Technology scene, an XML tag that allows an object to be clicked through so that an object behind receives the click.
polygon	3D scenes are drawn using polygons or triangles, vastly simplifying the computer creation of a 3D world. Triangles are defined as three coordinates-x, y, and z-one for each vertex.
preloader	A file that loads quickly to display while the other scene files are loading. Also, the name of the element used to define the order in which files are loaded. A file with a Preload priority of 1 downloads first, a priority value of 2 downloads second, and so on.
primitive	Standard geometric 3D objects such as a sphere, cube, box, cone, cylinder, and plane.
procedural	Equivalent to parametric. Procedural atoms, such as geometry and lightmaps, are referenced in an .mtx file and rendered based on preset values in Viewpoint Media Player. They add very little to a scene's overall file size.
progressive anti-aliasing	See *AccuMax*.
properties	Attributes of a media atom.
QTVR	QuickTime VR is Apple Computer, Inc.'s proprietary technology for creating cylindrical panorama and object movie files (.mov format) that allow viewers to be immersed in a 360° environment.
render	The transformation of 3D data into 2D frames for display on a computer screen.
root instance	By default, the highest-level parent object in an .mtx file.
rotation	Moving an object around a specific center and axis.
run-time	The period of time during which a program is executed.
scale	Changing the size of an object around a specific axis.
scene	The highest level of the Viewpoint Technology hierarchy (MTSScene tag in XML). Scene contains all elements of the .mtx and .mts files.

shadow blend	Toggles the shadows in the scene between shadows that blend with an environment and those that are projected onto the ground plane.
shadow opacity	A shadow's intensity: the higher the setting, the darker the shadow.
smoothing	See *smoothing groups*.
smoothing groups	An area on an object in which the rendering engine shows the polygons with a rounded surface of some degree. Can be used on a sphere to show a very rounded effect with much fewer polygons, instead of using a very large number of polygons to get the visual but not true effect of roundness.
soft body animation	See *morphing mesh*.
spline	A mathematical curve based on traditional shipbuilders' tools used to describe a hull shape. In computer animation, a spline is used to describe the motion path of an object through time and space. In modeling, a spline is a curve typically comprised of four control vertices used to control the shape of the geometry.
specularity	Defines an object's shiny highlights. When an object is rotated, how much it reflects depends on the material properties. The color of this reflection is defined by the specularity. Specularity determines how sharp or diffuse an edge the light has at the selected light point and sets the shininess at that point by determining how much light is reflected.
StartState	Defines the current state of the state machine (interactor) under which a transition (action) is available. The StartState may be omitted in the description of the interactor, if the transition is to be state independent.
state machine	Viewpoint Media Player 3.0.7 or greater implements a basic interaction interface as a state machine. A state machine is a collection of states for a scene, of which one is active (current). In addition, it is a transition map of how to get from one state to another based on input (events). When transitioning from state to state, an action can be associated with each state. A scene can contain more than one state machine.
sub-dividing	See *tessellation*.
SWF	The Macromedia Flash™ movie file extension. May be pronounced "swif."
tessellation	Decomposing a complex surface into a series of simple ones that approximate the complex surface. Determines the finesses of the segments making up the model. A higher tessellation results in a smoother model but larger file size. Scene Builder allows the user to define the surface of procedural geometry and allows the user to adjust the number of triangles that define that surface.
texture	A picture on the surface, usually a JPEG or similar image file. This image file is rendered over polygons to give the object a realistic-looking surface.

tiling	The method of repeating a texture more than once across an object or part of an object. A tiled texture looks best if its edges seamlessly match up with each other, top to bottom and side to side. Tiling is a common method of using the smallest texture possible to cover a large area, such as a texture of a brick tiling across a large polygon or object to create an entire brick wall.
transforms	Transforms are position, rotation, and scale.
translate	To move the object along the x, y, or z axis in the scene.
triangle	See *polygon*.
TrixelsNT	Viewpoint's proprietary image compression using wavelets.
tweening	Determines whether keyframes are interpolated.
UVW coordinates	Coordinates similar to XYZ that define the mapping of textures to polygons.
vertex	A point in 3D space that is connected to a polygon and the corner of a triangle where two lines meet. A triangle, therefore, is made of 3 vertices.
vertex translation animation	See *morphing mesh*.
Viewpoint Technology	Viewpoint Corporation's unique technology that streams 3D and rich media content (media atoms) over the Internet via Viewpoint Media Player, a web browser plug-in.
Viewpoint Media Player	The web browser plug-in necessary to view Viewpoint Technology content with Netscape Navigator or Internet Explorer.
wavelets	An image compression method.
widget	An area in a scene made up of a procedural shape (usually invisible) and created in Scene Builder. Widgets are generally used to define a hot spot that when interacted with displays a text annotation, texture, or Flash™ movie. For instance, when a widget is rolled over, a text annotation may appear.
wildcard	Represented by an asterisk (*). A symbol in Viewpoint XML that allows you to start an animation from the current state or position, rather than from a set position. This prevents jumping if the object has been moved from the desired starting point or if an animation is already playing.
wireframe	A representation of a 3D object that shows only the edges of its polygons.
XML	Extensible Markup Language. A markup language for documents containing structured information with instructions for content (words, pictures, etc.) and the role that content plays (for example, content in a section heading has a different meaning from content in a footnote, figure caption, or database table). Viewpoint Experience Technology uses XML to define all properties of a scene.
Zbuffer	A third buffer where depth data is stored that determines which textures are visible and which are hidden.