Reflectivity in materials architectural Visualisation

In this paper, the author will investigate Reflectivity in Architectural Visualisation. The author will be focussing on implementing the findings into a 3D environment within Autodesk Maya. The structure of this paper will consist of two main parts; a qualitative and quantitative study. The qualitative element will be created from information gathered from the literature review sourced from journal papers and internet sources. The second element; a quantitative study, will be formed by creating technical tests that will be shown to audiences to gather opinions and data in order to form opinions. The research will utilize questionnaires, surveys, and focus groups to gather evidence from the tests.

These two elements will then be compared, evaluated to form conclusions. These conclusions will be applied to a CGI BSc architectural visualization project. The author will use qualitative and quantitative processing methods in an academic triangulation method to process the results. The author will apply the findings of the paper to create a 120-150 second 3D CGI animation.

In this paper, the Author intends to research Reflectivity in Architectural Visualisation. The Author will conduct a qualitative and quantitative study and triangulate the information to find information and conclusions to the research questions. The author will look at and analyze current literature and previous studies that have been carried out in this field and this paper will identify key findings and discoveries. The Author will cross-reference previous and contemporary information to high areas of interest of concern. The second section of this paper will use quantitative research mechanisms to specifically test and evaluate key findings from the qualitative study. The author will then summarise the results and conclusions and areas for future research will be highlighted.

This research as stated will investigate reflectivity of materials within the field of 3D CGI architectural visualization animations. Before commencement can proceed it is important that the author provides definitions for the terms ‘reflectivity’ and ‘materials’ in CGI. Reflectivity is defined as “The property of reflecting light or radiation, especially reflectance as measured independently of the thickness of a material.” Oxford dictionary 2017. Date? Autodesk Maya has released 2 definitions of reflectivity one for smooth surfaces “Light bounces off the surface of a material at an angle equal to the angle of the incoming light wave.” Maya and one for hard surfaces “Light waves bounce off at many of angles because the surface is uneven” Autodesk Maya date?

Another important term ‘’materials’ within – quote the definition

To create a realistic computer-generated picture of a shiny surface it is often necessary to simulate the reflections in the surface. In the 1990's Ray tracing could provide accurate reflections, but required a great deal of CPU time. There were a few less time-consuming ways to simulate reflections back then using PRMan. However, None of the methods were both effective and efficient in all situations. For the best results, it was important that you chose the most appropriate method for your application.

The author will now describe a list of different methods that would have been used to simulate reflections. The first method uses a texture map and requires an additional rendering step to create the texture map from the scene.

There were less expensive methods for simulating reflections however the had a lower degree of realism. The sky is often the main source of reflections, in outdoor scenes. A simple shader that selects sky and ground colors based on the "up" component of the reflected vector would give an impression of reflections without the requirement of any additional rendering steps or texture files. This method worked well for surfaces that were curved but may have given less realistic results for large flat surfaces.

If the reflective surfaces were not flat, the technique described would not have worked. In such a case, reflections could have been simulated using environment textures. Additional rendering steps are needed to create the environment texture, and it took longer to render the final image. Reflections in curved surfaces may have been simulated quite accurately using environment maps, particularly if the reflected objects were not too close to the reflecting object. The environment map technique was far more realistic than the simple "sky and ground" reflection technique but was far more expensive.

Reflections in a plane: Simulating reflections was particularly easy when the reflecting surface was flat (planar).

Imagine that you point a camera at a mirror and take a picture of the image reflected in the mirror. Now imagine that the mirror is replaced with a clear glass window, and the camera is moved to an exactly opposite position on the other side of the window. Take the second picture from the new vantage point with the camera looking into the room through the window. Remarkably enough, when you compare the two pictures, one is the "mirror image" of the other, that is, the same image with left and right reversed. This thought experiment suggests a technique for simulating reflections in a mirror or other flat surfaces in a computer-generated picture.

In the mathematical world of computer graphics, we could simulate a reflection exactly (including the left-right reversal) by reflecting the camera through the mirror, instead of simply moving it to the other side of the mirror.

Reflecting the camera.

The camera used to render the reflection image was simply the scene camera reflected through the reflection plane. An example of this, the mirror lies in the plane ‘z=-0.05’. If the reflection plane were ‘z=0’ in world space, the camera could be reflected by adding the command: ‘RiScale (1., 1., -1.);’

This scale operation would do nothing except negate all of the z coordinates of the camera coordinate system. If the camera was positioned at (x, y, z) before the scale operation, it will be positioned at (x, y, -z) after the scale; this is, the reflection of the original position.

The case of reflection in the ‘z=-0.05’ plane is only slightly more difficult. First, we would translate the ‘z=0’ plane to the position of the actual reflection plane using a ‘translate’ call, then do the scale operation, which reflects through ‘z=0’, and then translate the ‘z=0’ plane back to its original position.

‘RiTranslate (0., 0, -0.5);’

‘RiScale (1., 1., -1.);’

‘RiTranslate (0., 0., 0.05);’

Notice that points which lie on the ‘z=-0.05’ plane in world space are unaffected by this sequence of transformations.

There is nothing special about z in the above procedure. The reflection through an ‘x=k’ or ‘’ y=k’ plane (for some number k) is very similar, simply negating the x or y coordinates using an appropriate RiScale call.

If the reflection plane were not aligned with the coordinate system axes, it's a little harder to reflect the camera through the reflection plane. Instead of using just a ‘translate’ call to move the ‘z=0’ (or ‘x=02’ or ‘y=0’) plane to coincide with the reflection plane, it was necessary to use both an ‘RiRotate’ and an ‘translate’. The axis of rotation is the cross product of the normal vectors of the reflection plane and the ‘z=0’ plane. The direction of translation is along the normal vector of the reflection plane. The inverse ‘RiTranslate’ and ‘RiRotate’ must be used to transform the reflection plane back to ‘z=0’ after the ‘RiScale’ is applied. If you prefer, all of the rotations, translations, and scales can be combined into a single transformation matrix that can be applied using the ‘RiConcatTransform’ call.

Rendering the reflective image.

If the example program was compiled with the preprocessor symbol REFLECTION defined, it would contain the RenderMan calls needed to render the reflection image that will appear reflected in the mirror when the final scene image is rendered. The reflection image is rendered into a TIFF file called ‘refl.tif’ and then used to make a texture file. It is most efficient to make the texture from a square image whose resolution is an integer power of 2. The scene image might not be square, so the pixel aspect ratio of the reflection image must be adjusted so that the reflection image covers the same screen window as the scene image. The screen window determines how much of the scene is visible in the image.

Making the Reflection texture.

Having rendered the reflection image, we could then make it into a texture file called refl.tex using the ‘RiMakeTexture’.

The ‘RI_BLACK’ parameters specify that the texture values outside the ‘0:1’ texture coordinate range of the reflection map should be zero. If the reflection texture is used properly, values outside the ‘0:1’ texture coordinate range should never be accessed, but the ‘RiMakeTexture’ call requires the specification of a wrap mode and ‘RI_BLACK’ is a reasonable choice. The filtering parameters ‘RiBoxFilter, 1., 1.’ are the default values for ‘RiMakeTexture’.

The Reflection shader.

The mirror or other reflecting surfaces must have a surface shader that uses the texture map created in the preceding step. Each pixel on the mirror is shaded using the texture map that was created from the reflection image. The pixel position in the scene image is used to look up the correct pixels from the texture map. In effect, the two images are being composited, but only at the pixels where the mirror is visible in the scene image.

Each point on the mirror is shaded with a color from the texture map, multiplied by the surface color and opacity so that colored and partially transparent mirrors are possible. The texture coordinates used to access the texture map are simply the x and y components of PNDC. PNDC is the point expressed in the NDC (normalized device coordinate) system, in which the x and y coordinates range from 0 to 1 across and down the image.

The simple shader shown here can be made more sophisticated by combining the reflection with a plastic shading model or a wood shading model. This would be appropriate to add reflections to a shiny floor or tabletop that is not a pure reflector like a mirror.