Study and Analysis of Texturing Techniques for Game Production

Sumeet Kalindi
09634006
Industrial Design Centre
IIT Bombay

Declaration

I hereby declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/ source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature: [signature]

Name of Student: Sumeet Kalindi
Roll No: 09634006
Date: 15/7/2011

Approval Sheet

The Design Seminar Research Project titled **“Study and Analysis of Texturing Techniques for Game Production”** by **Sumeet Kalindi (09634006)** is approved towards partial fulfilment of the requirements for the Post Graduate Degree of Masters of Design in Animation.

Guide : T. Phani
(PHANI TETALI)

Content


Assasins Creed ©Ubisoft


Medal of Honor 2010 © EA Games

Dirt 3 © Codemasters

Introduction

The major contributing factor to the engagement of a player in any console or computer game is part due to its convincing visuals. These visuals have to give a good approximation of the realism that the game is trying to represent. This realism is not subject to photorealism alone but even includes stylised or fantastical representation of reality. The creation of a texture is more than just defining the object colour and texture. It's about creating a sense of identity and history for the object. The texture shows the effect of time, weather and human interaction with the object. It defines the physical appearance that makes the object believable.

The surface details are added to the object by using images which define the respective surface properties. These images are termed as texture maps. The visual details of the computer generated models and environments are added by the use of texture maps in addition to the colour properties.

Texturing process is a very important part of game development and takes up to four times the time taken by modelling. Texturing helps in augmenting the visual perception of reality by trying to replicate the real world in the CG environment as faithfully as allowed by the hardware and software technologies. The paper aims to explore the texturing techniques as used in Game production.

Texturing

In *3D computer graphics* the process of adding details to the surface of a 3D object through image maps is called *texturing*. This image is called a *texture map*. In general when the 3D object is rendered, the texture map (image) supplies information to the renderer as to how the light shall behave on that particular surface. In context of games, texturing is adding information to the model's geometry which the game engine uses to redraw the model in the CG environment replicating how the object will react to light. So basically the texture defines the light reacting properties of the model geometry.

The surface properties of the object can be defined by the following main properties. The textures are also categorised roughly into these main categories.

Colour: Not all objects have uniform colour throughout their surface. There is a varying degree of uneven colour distribution in all real objects. A colour map defines these irregularities in colour distribution across the surface.

Diffuse: Diffusion is the attribute of a surface that scatters light. It determines how much of the surface colour is seen. It works well along with the colour map.

Specular: The specularity decides the shininess of the object which is different from glossiness.

Luminosity: Luminosity defines the self illuminating properties of the object.

Reflectivity: The reflectivity defines the glossiness of the object.

Refractivity: This defines the light bending properties of an object.

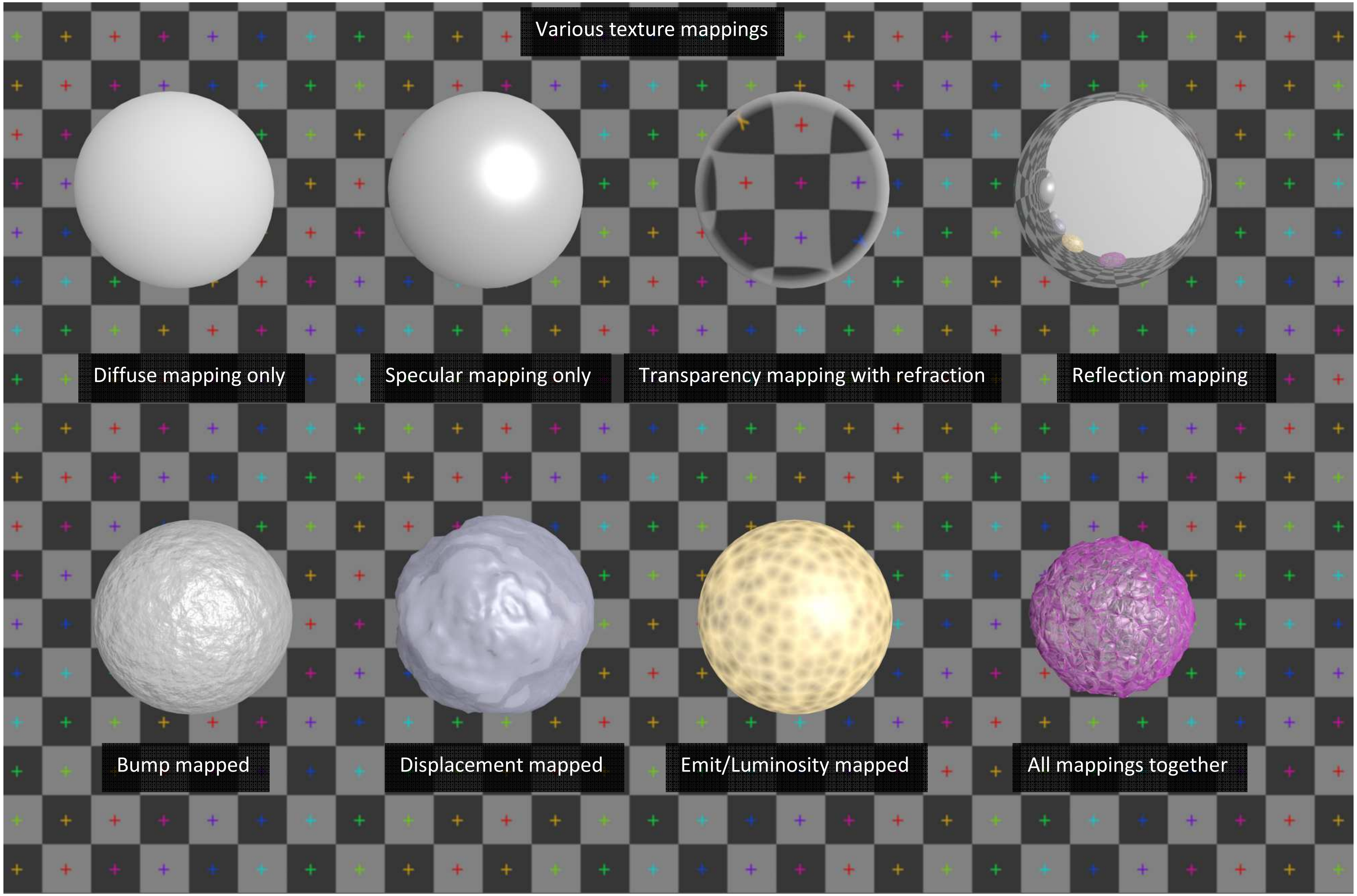
Transparency: Transparency defines the degree of light that the object allows to pass through it. Transparency mapping can also be used to cut out detailed shapes and patterns from a surface.

Translucency: Translucency helps define the ability of an object to be backlit without being transparent.

Bump: Bump map as the name suggests defines the roughness or bumpiness of the surface. Bump mapping works by changing the surface's shading as if small details had been present. The shading of a surface is based on an angle called the surface normal, which is usually perpendicular to the geometric surface of an object. A bump map changes the surface normals, making the object respond to light as if additional details had been present in the geometry.

Displacement: Displacement map is used to alter the geometry of the object based on height maps. The displacement map changes the shape of the object during rendering.

Normal: Normal mapping is similar to bump mapping in that it modifies the specular shading without changing the geometry. In a normal map, a 3D angle for a normal is directly specified by three values per pixel, stored as three colour channels in the map.


Various texture mappings
Diffuse mapping only
Specular mapping only
Transparency mapping with refraction
Reflection mapping
Bump mapped
Displacement mapped
Emit/Luminosity mapped
All mappings together

Texel – a *texel,* or *texture element* (also *texture pixel*) is the fundamental unit of texture space. Textures are represented by array of texels, just as an image is represented as an array of pixels.
Shader – *Shader* is a set of software instructions that is used primarily to calculate rendering effects on graphics hardware with a high degree of flexibility.

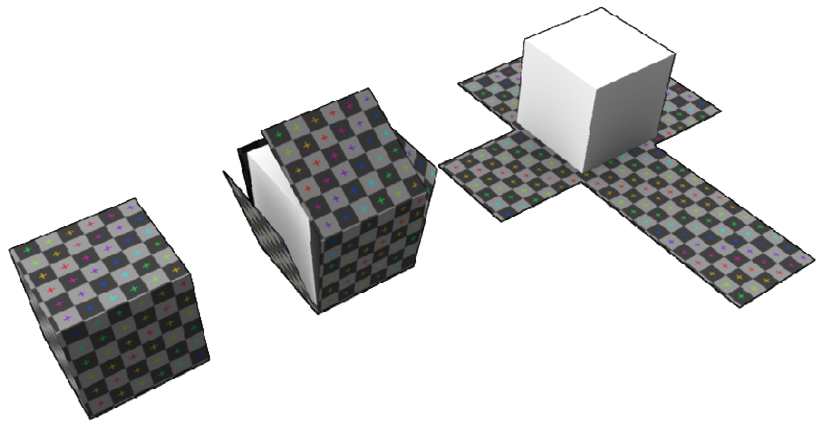
Material based texturing – Material based texturing gives a property that applies to the surface uniformly. Image maps and how they wrap around the object is not required to be done. Hence they are easier to create and implement.

Image based texturing – In image based texturing the surface properties are defined by various image maps that are applied to the geometry of the 3D object. The textures take up video memory when they are loaded onto the system for rendering.

Texture Mapping

A *texture map* (image) is applied (mapped) to the surface of a shape or polygon. Every vertex in a polygon is assigned a texture coordinate (which in the 2d case is also known as a UV coordinate) either via explicit assignment or by procedural definition. *Multi-texturing* is the use of more than one texture at a time on a polygon.

UVW mapping – UVW mapping is a mathematical technique for co-ordinate mapping which is used to convert 2d images to map onto a three dimensional topology. UVW, like the Cartesian co-ordinate system has 3 dimensions, where UV dimensions define the image plane and W dimension helps the image to wrap onto irregular or complex surfaces. Each point on the UVW map corresponds to a point on the surface of the object. UVW mapping is the choice for texturing objects that have non-primitive shapes or complex geometries

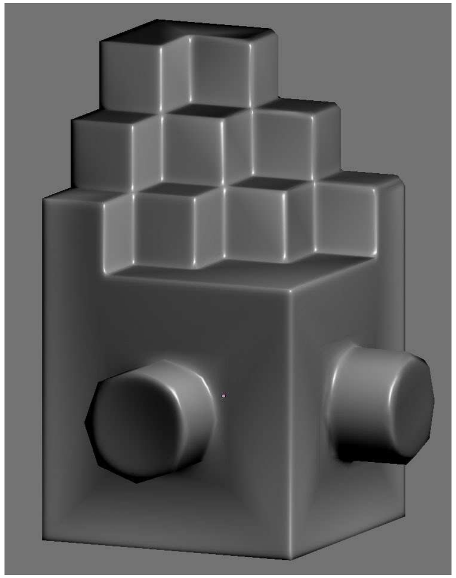


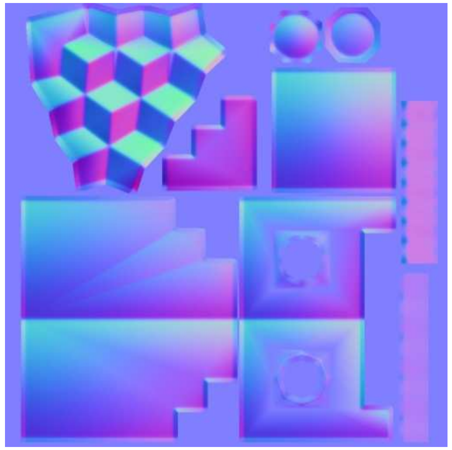
Procedural texture – Procedural texture is a texture generated by an algorithm which tries to depict natural objects realistically like wood, granite, marble, etc.

Bump mapping - Bump mapping is a technique to make a rendered surface look more realistic by modelling the interaction of a bumpy surface texture with lights in the environment. Bump mapping does this by changing the brightness of the pixels on the surface in response to a heightmap that is specified for each surface. When rendering a 3D scene, the brightness and colour of the pixels are determined by the interaction of a 3D model with lights in the scene. After it is determined that an object is visible, trigonometry is used to calculate the "geometric" surface normal of the object, defined as a vector at each pixel position on the object. The geometric surface normal then defines how strongly the object interacts with light coming from a given direction using a lighting algorithm. Light travelling perpendicular to a surface interacts more strongly than light that is more parallel to the surface. The bump mapping algorithm also ensures that the surface appearance changes as lights in the scene are moved around. Normal mapping is a commonly used variant of the basic bump mapping technique, another variant is parallax mapping. An important thing to take into consideration is that silhouettes and shadows therefore remain unaffected since the underlying geometry is not changed.

Normal mapping – Normal mapping is a technique used for faking the lighting of bumps and dents in an object to add the surface details without the need to add more polygons to the geometry. A normal map is usually an RGB image that corresponds to the X, Y, and Z coordinates of a surface normal from a more detailed version of the object. A common use of this technique is to greatly enhance the appearance and details of a low polygon model by generating a normal map from a high polygon model.

Normal mapping in Video Games – Normal mapping's popularity for real-time rendering is due to its *good quality* to *processing requirements* ratio versus other methods of producing similar effects. Much of this efficiency is made possible by distance-indexed detail scaling, a technique which selectively decreases the detail of the normal map of a given texture (cf. mipmapping), meaning that more distant surfaces require less complex lighting simulation.





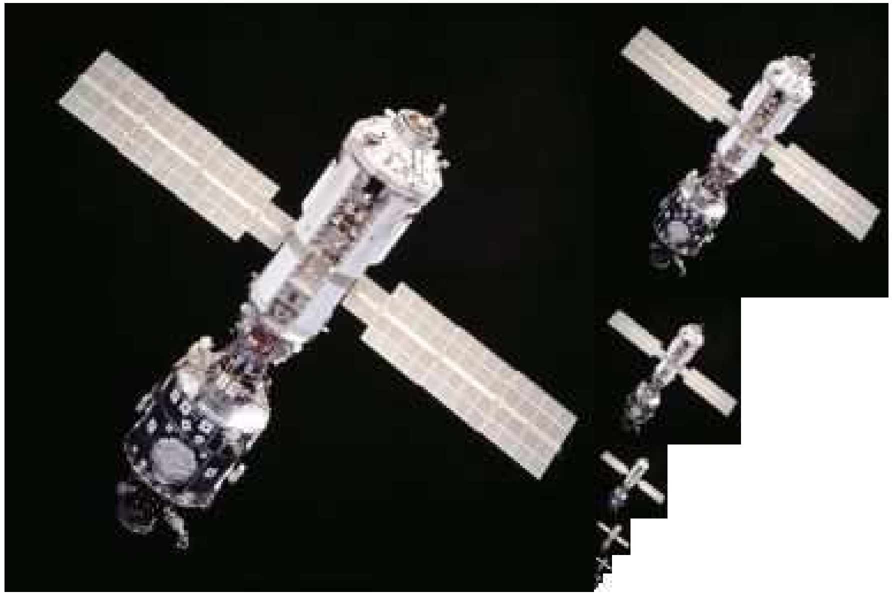
A model with its normal map

The Basic normal mapping can be implemented in any hardware that supports palettized textures. The first game console to have specialized normal mapping hardware was the Sega Dreamcast. However, Microsoft's Xbox was the first console to widely use the effect in retail games. Out of the sixth generation consoles, only the PlayStation 2's GPU lacks built-in normal mapping support.

Parallax mapping – Parallax mapping is implemented by displacing the texture coordinates at a point on the rendered polygon by a function of the view angle in tangent space (the angle relative to the surface normal) and the value of the height map at that point. At steeper view-angles, the texture coordinates are displaced more, giving the illusion of depth due to parallax effects as the view changes. Parallax mapping is an enhancement to bump mapping and normal mapping techniques.

Texture filtering – *texture filtering* is the method used to determine the texture colour for a texture mapped pixel, using the colours of nearby texels.

MIP maps – MIP maps (also **mipmaps**) are pre-calculated, optimized collections of images that accompany a main texture, intended to increase rendering speed and reduce aliasing artifacts. They are widely used in 3D computer games, flight simulators and other 3D imaging systems. The technique is known as *mipmapping*. The letters "MIP" in the name are an acronym of the Latin phrase *multum in parvo*, meaning "much in a small space". Mipmaps need more space in memory. Each image in a mipmap set is a version of the main texture but at a certain reduced level of detail. Rendering speed increases since the number of texture pixels ("texels") being processed can be much lower than with simple textures. Artifacts are reduced since the mipmap images are effectively already anti-aliased, taking some of the burden off the real-time renderer.



Texturing for games

Current generation games come in a variety of platforms from dedicated consoles, mobile devices to PC systems. The hardware in the dedicated consoles is more powerful than the hardware in other devices. In case of 3D games the models are rendered in real time within the game engine. The level of detail that is being incorporated into the games is increasing by the day. Complex surfaces and geometries are required to represent these geometries. But even with very fast computing machines be it consoles or personal computer, the amount of data that needs to be calculated to recreate the virtual models in real time is phenomenal.

The render time in games is heavily dependent on the amount of calculations that need to be done to reproduce the image. This data includes calculations of the geometry of the model (the actual no of vertices or polygons that define the model), the calculation of light interactions with the model (includes colour, texture, etc), and the other interactivity inherent in the game. The textures used in the games can greatly reduce render time by cutting down the calculations required for how the model geometry interacts with light or the surface properties.

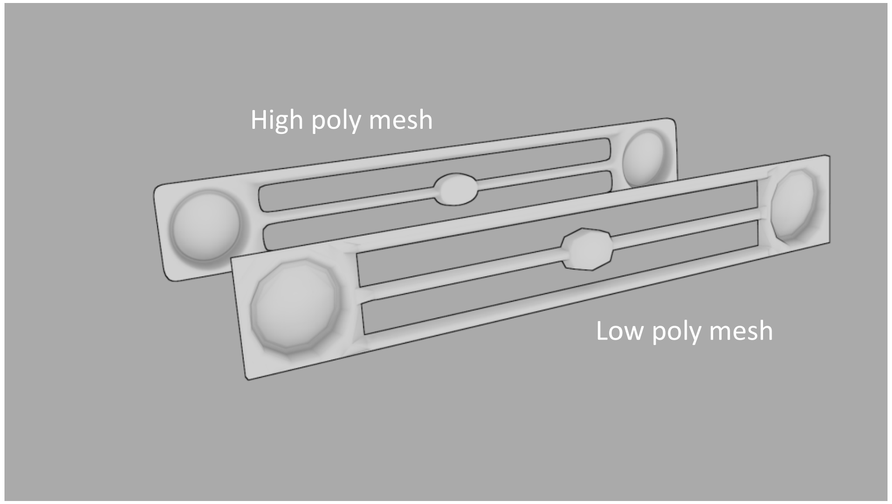
One of the main domains of engagement in a game is the emotional atmosphere it creates for the player. This atmosphere is heavily dependent on the visual quality. The visuals have to be convincing in a sense that they are believable even if they might not be existing in the real world. Thus realism is not completely imply photorealism but includes objects that may be unrealistic but convincing.

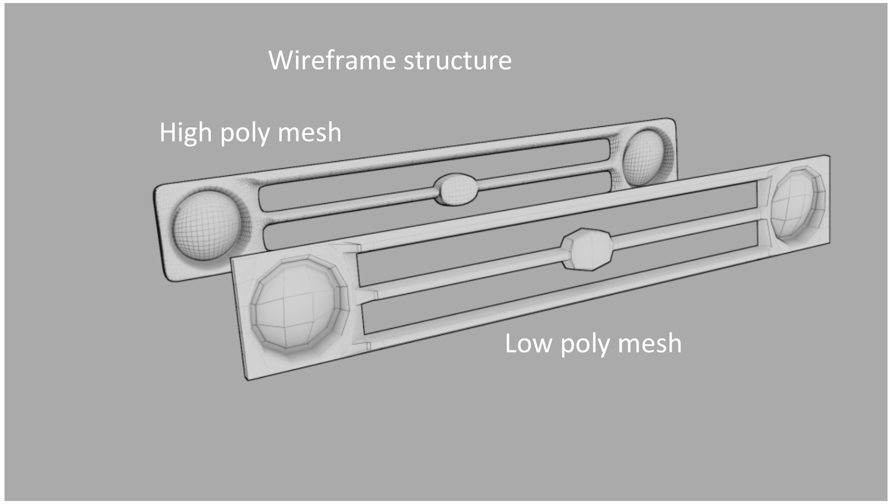
Textures help create this realism but replicating how light behaves with an object. The texture gives the object its surface properties which gives a sense of believability to the 3d object.

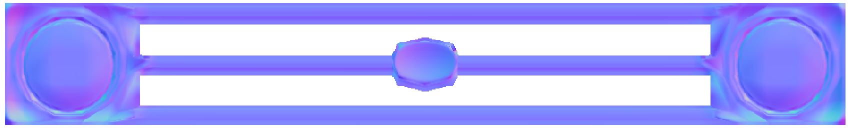
Today *Graphical Processing Units* (GPUs) or Graphics cards can perform parallel computing faster than a normal CPU. The rendering of the virtual environment is done by the GPU and the other calculations are done by the CPU. The main requirement in games is that the 3D environment or the models are rendered in real time. All the light calculations or ray tracing is done beforehand and saved as textures. The ray tracing method of rendering is used to get a life-like approximation of light interactions with the models and environment. Ray tracing is a very time consuming and processor exhaustive render method but yields very good results. But ray tracing cannot be incorporated in a game engine because of the real time render constraint. Hence the method of saved textures mapped onto the models is used. Nowadays DirextX and OpenGL have taken off a bit of load of the hardware based rendering.

The general workflow of adding textures is having separate render passes for each of the surface attributes and baking them to a UVW mapped image. This UVW image is then mapped onto the 3D model into separate texture channels according to the surface attribute. While the game engine is rendering the model it loads the pre-rendered texture images and maps them to the model geometry. Here the amount of video memory is very critical for the textures to be loaded. For tiled textures usually have a size varying from 16 x 16 pixels to 512 x 512 pixels. Nowadays even higher texture sizes are possible. The size of the texture is also dependent on the amount of details required on the model. For distant objects the size can be smaller but the size required increases as the model comes closer to the camera. For a full profile of the model covering the screen, the size should be at least equal to the height of the screen resolution in pixels. Commonly in such cases the height is taken to be twice the height if the screen resolution. The consoles have a higher hardware configuration than the personal computers since they are dedicated components specific to graphics processing. The texture sizes in consoles are usually higher than in personal computers.

Normal mapping example







Normal mapping

This example was part of an exercise to look at how normal mappings affect the geometry of the model. A vehicle was modelled and this part is the front head lamp and air intake grill (grill not shown). The High polygon mesh was used to get the details into a low polygon version. The basic structure of both the polygon meshes need to be similar if not same. As can be seen in the wireframe render that the number of polygons is smaller in the low polygon mesh and it still retains the details of the high polygon mesh. The smoothness of the low polygon mesh can be further enhanced by adding specular, bump map and reflectance map to name a few. The shadows and highlights added due to the maps will fake an approximation of the actual high polygon surface.

Bibliography

1. **Introduction to Low-Polygon Game Asset Texturing**

   homepage.mac.com/nephilim/sw3ddev/texturing_tutorial.pdf

2. Andrew Glassner, **An Introduction to Ray Tracing**, San Francisco: Morgan–Kaufmann, 1989
3. **Texture Mapping**, *http://en.wikipedia.org/wiki/Texture_mapping*
4. **Digital Lighting & Rendering,** 2nd Ed., Jeremy Birn
   ISBN : 978-0-321-31631-8
5. **Bump mapping**, *http://en.wikipedia.org/wiki/Bump_mapping*
6. **Normal mapping**, *http://en.wikipedia.org/wiki/Normal_mapping*
7. **Texturing concepts and Techniques,** Dennis Summers
   ISBN : 1-58450-300-9
8. **Blender texture mappings**,
   http://wiki.blender.org/index.php/Doc:Manual/Render/Bake