matrox Parhelia™

High Fidelity Graphics™

Hardware Displacement Mapping

matrox

matrox.com/mga

Hardware Displacement Mapping

Matrox's revolutionary new surface generation technology, Hardware Displacement Mapping (HDM), equates a giant leap in the pursuit of 3D realism. Matrox is the first to develop a hardware implementation of displacement mapping and has contributed elements of this technology for inclusion as a standard feature in Microsoft®'s DirectX® 9 API. Introduced in the Matrox Parhelia™-512 GPU, Hardware Displacement Mapping enables developers of 3D applications, such as games, to provide consumers with a more realistic and immersive 3D experience.

"Matrox's Hardware Displacement Mapping is a unique and innovative new technology that delivers increased realism for 3D games," **said Kenny Mitchell, director 3D graphics software engineering, Westwood Studios®.** *"The fact that it only took me a few hours to get up and running with the technology to attain excellent results was really encouraging and I look forward to integrating the technology more widely in future Westwood titles."*

Introduction

The challenge of creating highly realistic 3D scenes is fuelled by our desire to experience compelling virtual environments—whether they form part of 3D-rendered movies or interactive 3D games. How well a 3D-rendered scene reflects reality depends on the accuracy of its shapes, or **geometry,** in addition to the fidelity of its colors and **textures.**

The geometry of objects in a 3D scene is defined by 3D **meshes,** which are sets of connected triangles that represent the surface of these objects. To create 3D scenes with greater detail and realism, the number of triangles needed to represent objects within the scene must increase at an exponential rate. Consequently, rendering scenes of such complexity requires a technology that not only enables the dynamic generation of complex geometry, but also allows the high-resolution detail of such geometry to be encoded and stored using a simple and compact data representation.

Hardware Displacement Mapping (HDM), a powerful surface generation technology, combines two Matrox initiatives—**Depth-Adaptive Tessellation** and **Vertex Texturing**—to deliver extraordinary 3D realism through increased geometry detail, while providing the most simple and compact representation for high-resolution geometric data.

Unlike many other tessellation or higher-order surface generation techniques, Hardware Displacement Mapping does more than simply create smoother curves; it actually extrudes real geometric detail. Hardware Displacement Mapping can be applied to static environments, such as terrains, as well as to dynamic skinned objects, such as realistic 3D characters. Additionally, HDM provides a unique and powerful method of generating geometry with dynamic level of detail **(LOD),** which maximizes overall scene detail by increasing it in some areas and reducing unnecessary detail in others.

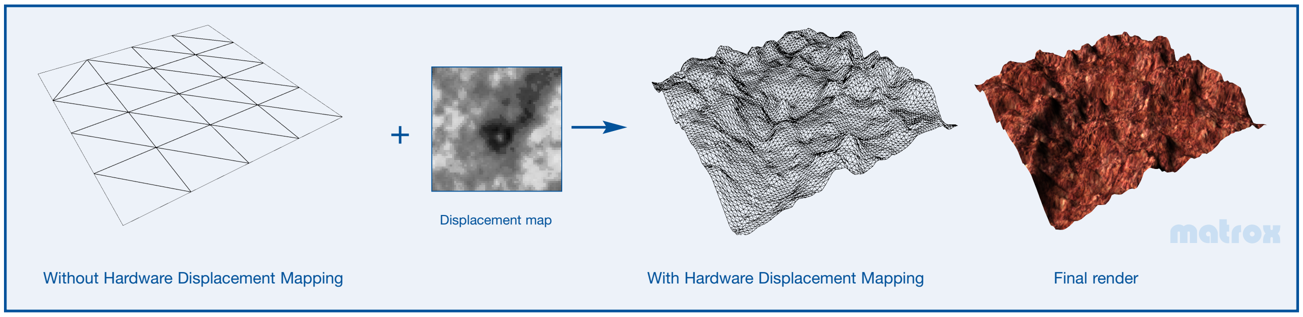

+
Displacement map
Without Hardware Displacement Mapping
With Hardware Displacement Mapping
Final render
matrox

Figure 1
Hardware Displacement Mapping provides higher detail 3D rendering.

matrox
matrox.com/mga

Hardware Displacement Mapping

What is displacement mapping?

Although the theoretical concept of displacement mapping is well-known in the graphics industry, it is important to provide some background information on this concept before explaining its first-ever hardware implementation.

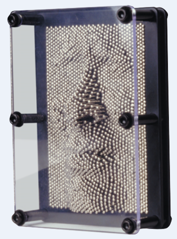
Theoretically, displacement mapping is a technique that uses **displacement maps** to displace the vertices of a mesh. The resulting **displaced mesh** takes the shape represented by the displacement map.

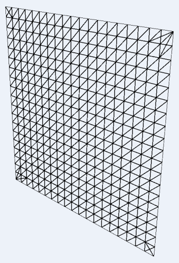
Similar to texture maps, displacement maps are 2D bitmaps; however, the individual values of a displacement map represent height information instead of texture colors **(texels).** Collectively, these height values define the shape of the displaced mesh surface. When mapped onto the vertices of the initial mesh, the displacement map displaces the vertices to form the displaced mesh.

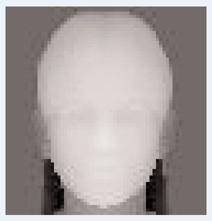
The concept of displacement mapping is similar to the effect produced by a bed of nails. A face pressed against a bed of nails acts like a displacement map, providing height data for each nail. When the face is mapped onto the nails, the nails are displaced, and mimic the shape of the face.

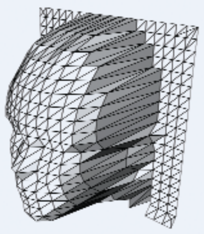
Flat bed of nails

Object


Displaced bed of nails


Source mesh


Displacement map


Displaced mesh

matrox.com/mga

Enhancing displacement mapping using tessellation

The concept of displacement mapping and its benefits can be extended by using displacement mapping in conjunction with **tessellation**—a process that converts a mesh with a low triangle count into a mesh with a higher triangle count.

While tessellation does not significantly alter the underlying shape of the object, it sometimes helps to smooth the surface of the object, much the same way that N-Patch tessellation does (figure 2). Unlike tessellation, displacement mapping can alter the shape of the underlying mesh considerably, based on data in the displacement map. By mapping the values of a displacement map onto the vertices of a tessellated mesh, the displaced mesh has both the desired shape and a much smoother surface.

Matrox has developed and defined the hardware implementation of displacement mapping, which combines unique tessellation technology with texture mapping on geometry, to raise the bar for 3D realism.

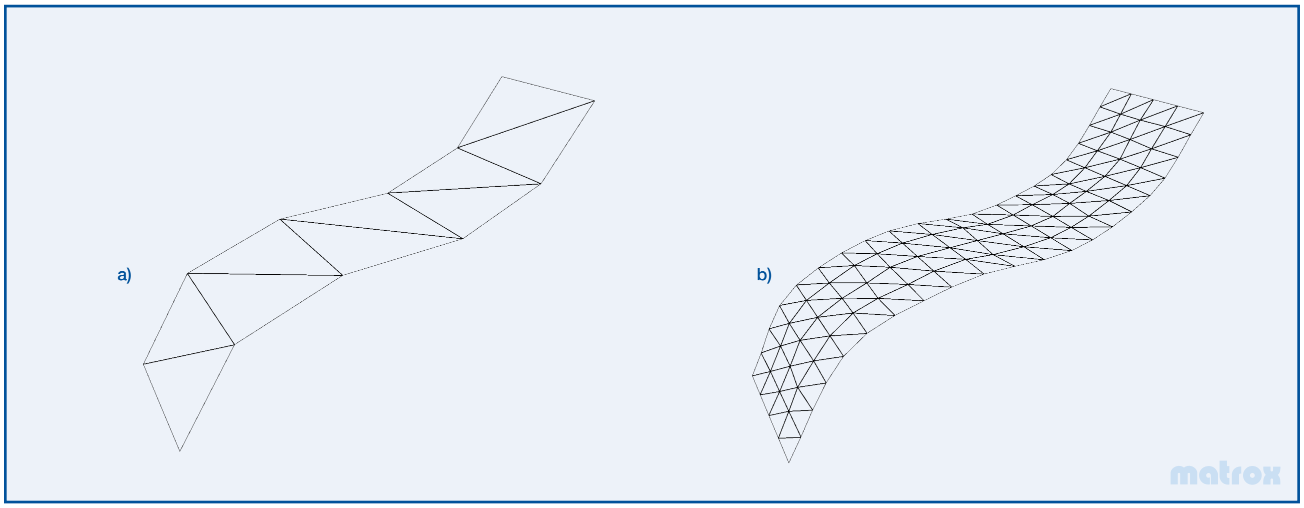


Figure 2
A mesh with a low triangle count **(a)** looks significantly smoother when tessellated **(b)**.

Hardware Displacement Mapping

The process of Hardware Displacement Mapping

As illustrated in figure 3, Hardware Displacement Mapping uses a low-triangle mesh, or a **base mesh,** in conjunction with a displacement map to produce a smooth and detailed mesh that can subsequently be textured or shaded using vertex and pixel shaders. This process can be divided into three stages:

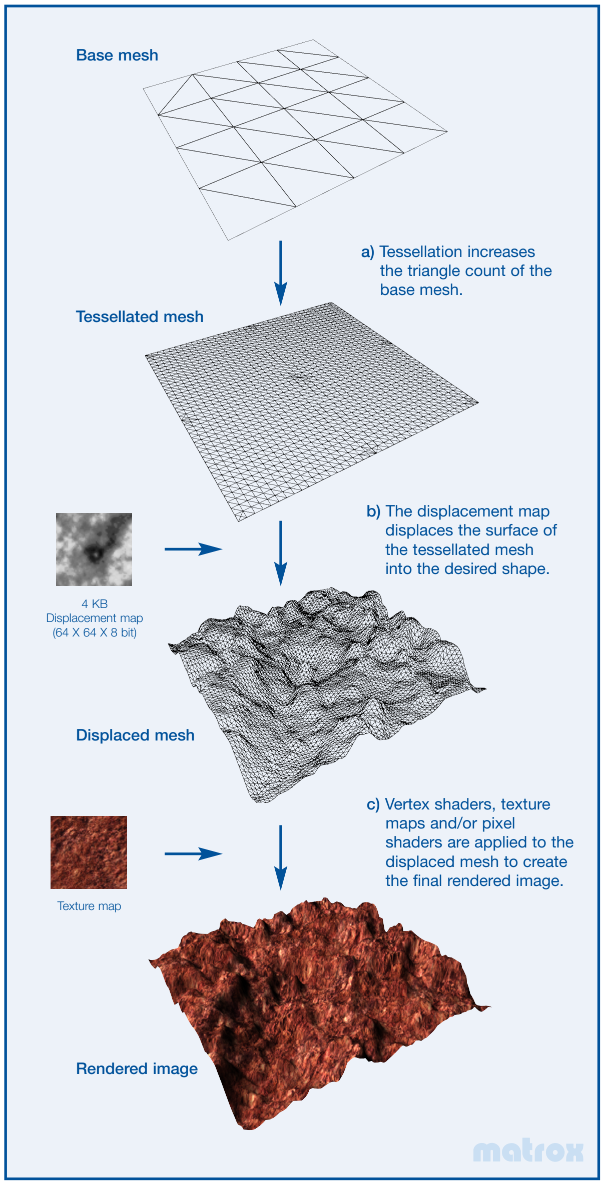

Base mesh
a) Tessellation increases the triangle count of the base mesh.
Tessellated mesh
b) The displacement map displaces the surface of the tessellated mesh into the desired shape.
4 KB Displacement map (64 X 64 X 8 bit)
Displaced mesh
c) Vertex shaders, texture maps and/or pixel shaders are applied to the displaced mesh to create the final rendered image.
Texture map
Rendered image
matrox

Figure 3
The process of Hardware Displacement Mapping.

Stage 1: Base mesh tessellation

A base mesh and its associated displacement map are sent to the GPU over the AGP bus. The GPU tessellates the base mesh to increase its triangle count (figure 3a). The base mesh can be of any level of detail and is compatible with any existing or legacy hardware.

As part of its Hardware Displacement Mapping technique, Matrox has implemented an innovative approach to tessellation called **Depth-Adaptive Tessellation.** This patent-pending tessellation scheme allows for varying levels of detail across a mesh and maintains smooth transitions between the levels of detail (LODs) without any geometry popping or cracking artifacts. Depth-Adaptive Tessellation is explained in detail on page 6.

Stage 2: Mapping textures onto geometry

Once the base mesh has been tessellated, the position of each vertex of the tessellated mesh is displaced (moved) in the direction of its vertex normal (figure 3b). The amount of displacement is determined as a function of the individual pixel values of the displacement map. The new position of each vertex is determined by mapping an associated value from the displacement map onto the vertex using a new concept by Matrox called **Vertex Texturing.** Vertex Texturing, explained on page 8, is similar to **texture mapping,** where a texture value, or texel, is mapped from a **texture map** onto each pixel of a triangle to determine the colors of its interior pixels. Displacement maps can be applied on both static base meshes and dynamic, skinned base meshes.

Stage 3: Rendering

Once the displaced mesh has been created, it is treated and rendered like any other vertex stream. Therefore, vertex shaders, textures and pixel shaders can be applied to the mesh to form the final rendered image (figure 3c).

Hardware Displacement Mapping

Depth-Adaptive Tessellation

Depth-Adaptive Tessellation is an advanced tessellation scheme that tessellates meshes using multiple levels of detail (LODs) to maximize the geometry detail of a 3D scene while maintaining high levels of performance (figure 4). By using LOD-based tessellation, the graphics processor avoids unnecessarily processing triangles that would otherwise not contribute significantly to the visual quality of the final rendered image.

Lower levels of detail are acceptable when the object being rendered is further back in the scene. Because it appears smaller, it is rendered using a lower number of screen pixels. In fact, depending on the distance, increasing the number of triangles beyond a certain point may have little or no effect on an object's appearance (figure 4c). The ability to reduce the LOD for distant objects provides considerable savings on transformation, lighting, setup and rasterization, leaving a higher triangle budget for objects that are up-close.

LODs can also be applied to an object whose mesh spans a significant portion of the scene's depth on the display. A good example of such an object is a terrain. For such objects, Depth-Adaptive Tessellation ensures that no cracks are seen on the seams where changes in LODs occur.

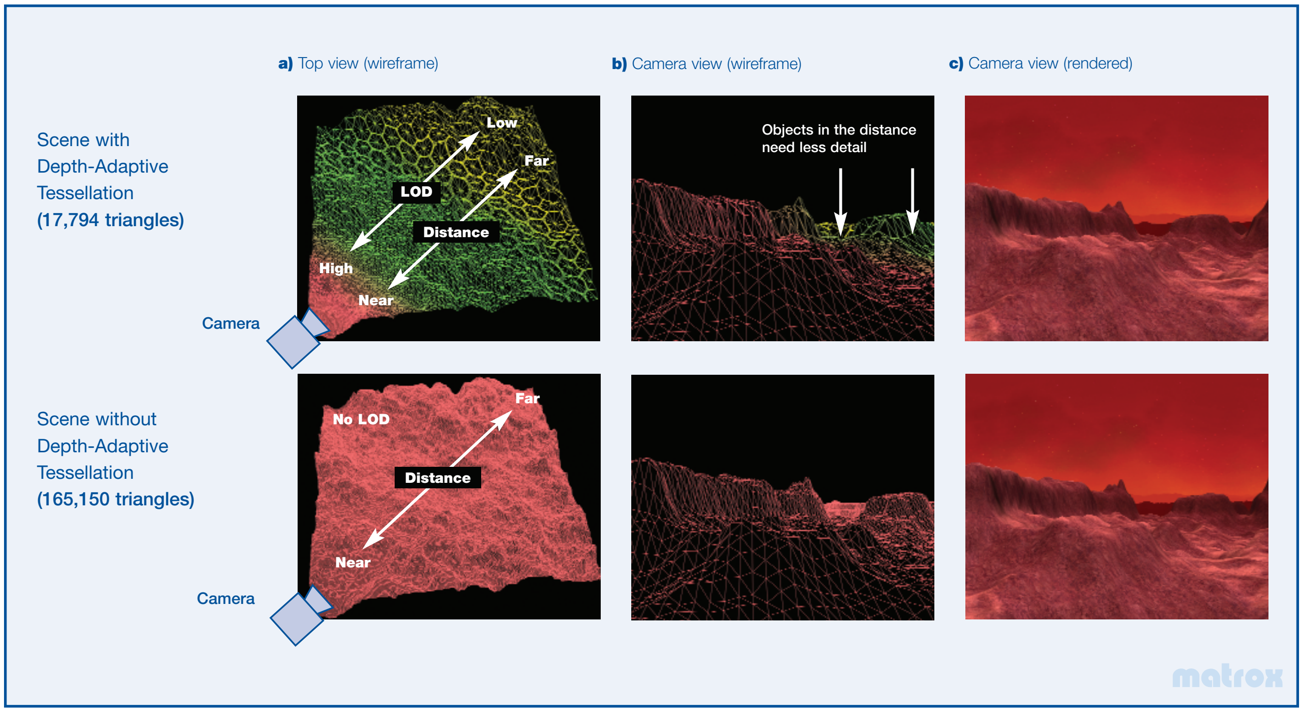

a) Top view (wireframe)
b) Camera view (wireframe)
c) Camera view (rendered)
Scene with Depth-Adaptive Tessellation **(17,794 triangles)**
Low
Far
LOD
Distance
High
Near
Camera
Objects in the distance need less detail
Scene without Depth-Adaptive Tessellation **(165,150 triangles)**
No LOD
Far
Distance
Near
Camera
matrox

Figure 4
a) Top view of the entire terrain with varying LODs with respect to the camera. **b)** The portion of the terrain in the distance is rendered using a lower level of detail. Depth-Adaptive Tessellation dramatically reduces the number of triangles required in the scene while maintaining high geometry detail during rendering. **c)** The difference in rendering between the two scenes is not noticeable.

matrox.com/mga

Depth-Adaptive Tessellation also allows for the dynamic variation of an object's level of detail (LOD) as it moves closer to, or further away from, the foreground. The LOD determines the **tessellation rate** or amount of tessellation for each triangle in the mesh, based on one or a combination of several factors, such as distance. Each triangle is tessellated independently into the appropriate number of sub-triangles using its tessellation rate. As the position of the object changes onscreen, the tessellation rate of each triangle, or groups of triangles, may change frequently. By supporting floating-point (non-integer) tessellation rates (1.2, 2.5, etc.), Depth-Adaptive Tessellation ensures that each triangle's tessellation rate changes smoothly (figure 4a). Without support for floating-point tessellation rates, the appearance of the object can change abruptly as portions of the object move from one level of detail to another—this results in artifacts referred to as **popping.**

By providing smooth transitions between LODs, Depth-Adaptive Tessellation ensures that adjacent triangles with different tessellation rates do not have cracks or exhibit undesirable popping artifacts during dynamic tessellation (figure 4b). Highly flexible and scalable, Depth-Adaptive Tessellation is compatible with most higher-order surface types, including N-Patches. Moreover, Depth-Adaptive Tessellation has the added advantage of providing the best tessellation environment for displacement mapping, as explained in the next section. A patent-pending technology, Depth-Adaptive Tessellation offers the highest quality while providing maximum scalability, efficiency and performance.

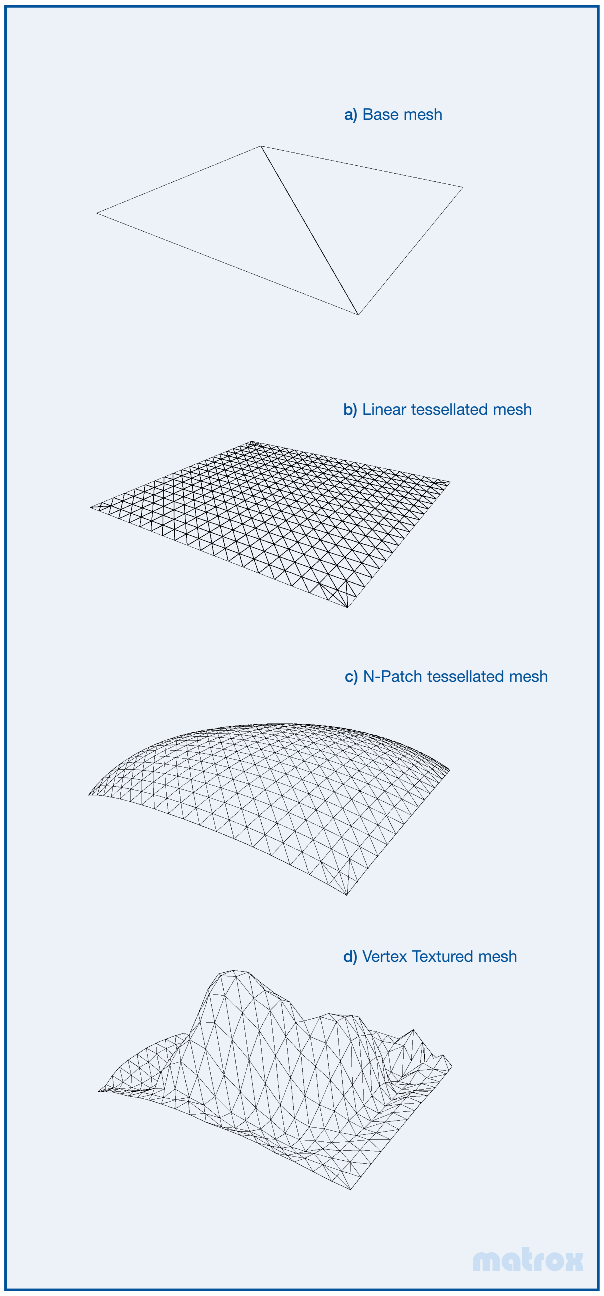


Figure 5
Vertex Texturing generates real geometric extrusions.

Mapping textures to geometry using Vertex Texturing

Vertex Texturing is a key aspect of displacement mapping. While the tessellation of a base mesh generates new vertices and their normals, the position of these new vertices can be calculated only by interpolating between the positions of the base vertices. Therefore, tessellation on its own cannot alter the underlying shape of the base mesh (figure 5b). Some higher-order surface descriptors, such as N-Patches (also known as PN triangles), can add curvature to a tessellated mesh to smooth the surface of an object. However, these descriptors cannot add any arbitrary detail to the mesh (figure 5c). In contrast, Vertex Texturing has no such limitations. By mapping textures onto geometry, the positions of the new vertices can be defined by the values or different functions of the values in the texture (figure 5d). In fact, these values can even be mapped onto the vertices after an N-Patch has been applied.

Hardware Displacement Mapping is one of the most compelling features enabled by Vertex Texturing. In the case of Hardware Displacement Mapping, the texture is referred to as a displacement map. Similar to texture mapping, each vertex (pixel) is associated to the displacement map (texture map) by means of a displacement texture coordinate (texture coordinate). Displacement texture coordinates, like texture coordinates, are assigned using a "u,v" coordinate system.

Displacement maps and filtering

Like texture mapping, a one-to-one correspondence between a vertex (screen pixel) and a displacement value (texel) is uncommon. Therefore, the displacement map needs to be up-sampled (up-scaled) or down-sampled (down-scaled) when it is mapped onto its associated vertices. When scaling textures for texture mapping, the use of various filtering techniques, such as bilinear and trilinear filtering, is common. Similarly, these filtering methods are used with displacement maps to calculate the most appropriate displacement value for each vertex.

Displacement maps and mip-mapping

When mapping the same texture onto triangles with different resolutions (numbers of pixels), it is common to have that texture available in different resolutions, or **mip-maps.** Mip-maps are pre-filtered, downscaled versions of the original texture.

During texture mapping, texels are fetched from the mip-map with the nearest resolution, so that very little up-sampling or down-sampling of the texture is required. This process is referred to as **mip-mapping.** Similarly, as shown in figure 6, displacement mapping accommodates dynamic level of detail (LOD) tessellation of meshes by mip-mapping the displacement maps: a mesh or the portion of a mesh with low level of detail will use a lower mip-map of the displacement map and vice versa. Mip-mapping of displacement maps is especially useful for Depth-Adaptive Tessellation, where the level of detail of a mesh can vary across different portions of that mesh. When this is the case, trilinear filtering can be applied on the two adjacent mip-maps of the displacement map when mapping the displacements. This ensures that no surface jittering occurs on the mesh where changes in mip-map levels occur.

The ability to use mip-mapping and advanced filtering methods with displacement maps makes Hardware Displacement Mapping one of the most powerful and scalable surface generation techniques available. Moreover, this technique provides a convenient way of generating procedural geometry by simply modifying a displacement map at runtime (procedural displacement mapping).

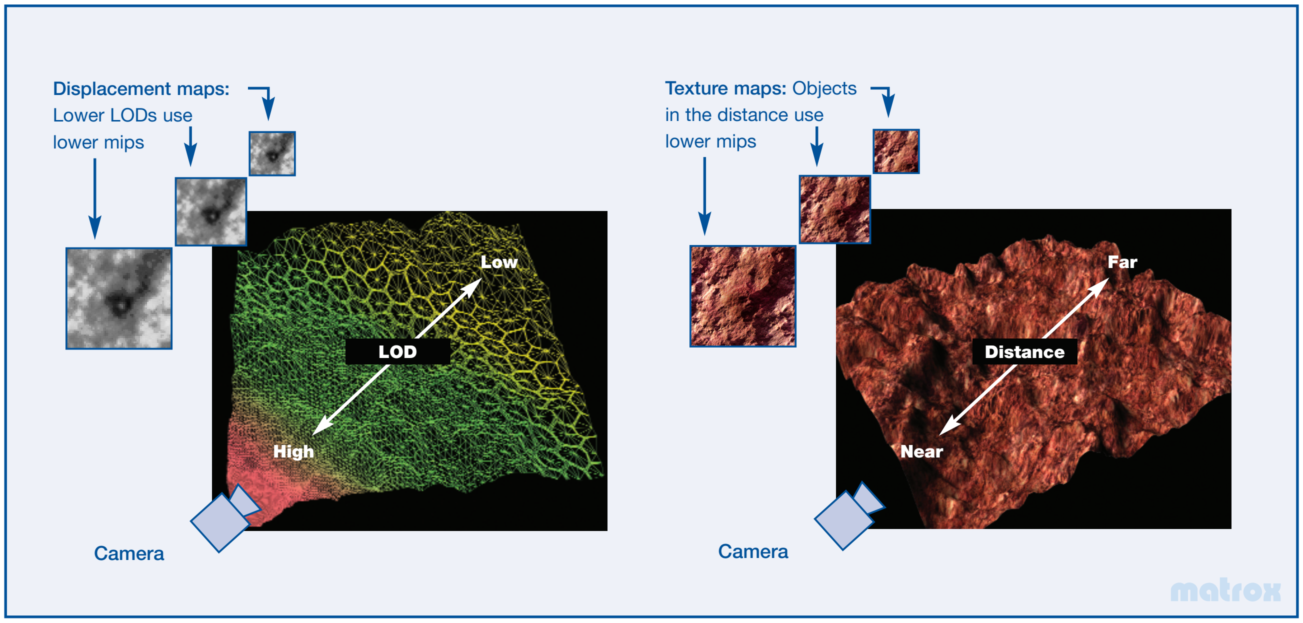

Displacement maps: Lower LODs use lower mips
Low
LOD
High
Camera
Texture maps: Objects in the distance use lower mips
Far
Distance
Near
Camera
matrox

Figure 6
Mip-mapping of displacement maps is similar to mip-mapping of texture maps.

Versatility of Hardware Displacement Mapping

- Hardware Displacement Mapping works with both static and skinned meshes. Figure 7 shows how displacement mapping can be used with skinned characters to provide high levels of detail on characters. Moreover, different displacement maps can be used on the same base mesh to render distinctly different objects. Hardware Displacement Mapping fits seamlessly into the 3D graphics pipeline and is fully compatible with vertex shaders and pixel shaders. Effects using vertex and pixel shaders can easily be layered onto displacement-mapped geometry.

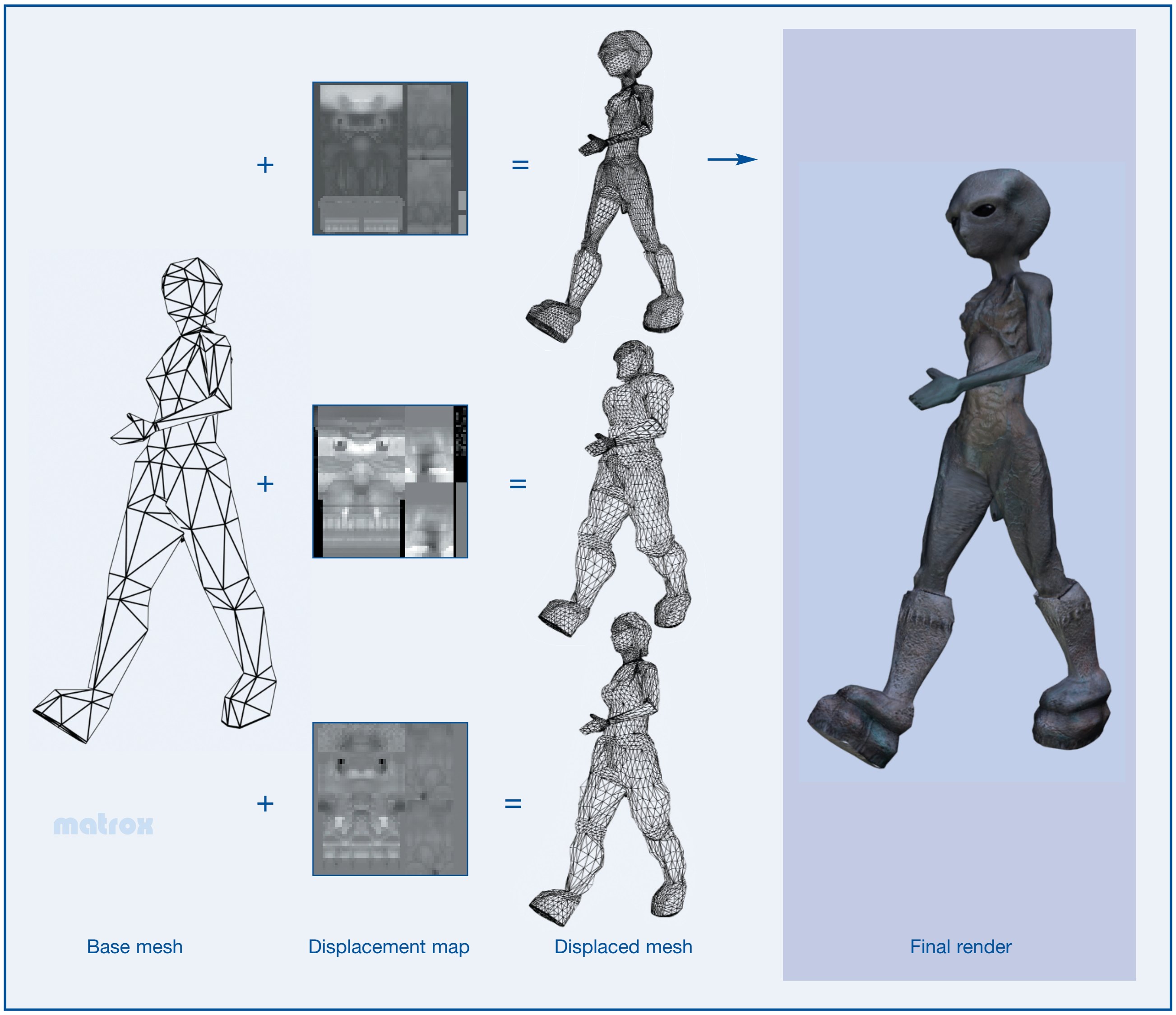


Figure 7
Displacement mapping is also suitable for objects with dynamic skinned meshes such as characters.

matrox.com/mga

Hardware Displacement Mapping

Due to the fact that displacement mapping uses 2D bitmaps to specify the shapes of objects, a large amount of readily available data can be used to generate new 3D objects and scenes in applications—especially games. As an example, U.S. Government geological survey data is stored as elevation information that can be converted directly to displacement maps. This geological survey data represents the topography of regions and terrains around the world and can be used to create virtually identical 3D terrains. Figure 8 shows an actual geological map of the Grand Canyon and its 3D version generated using Hardware Displacement Mapping.

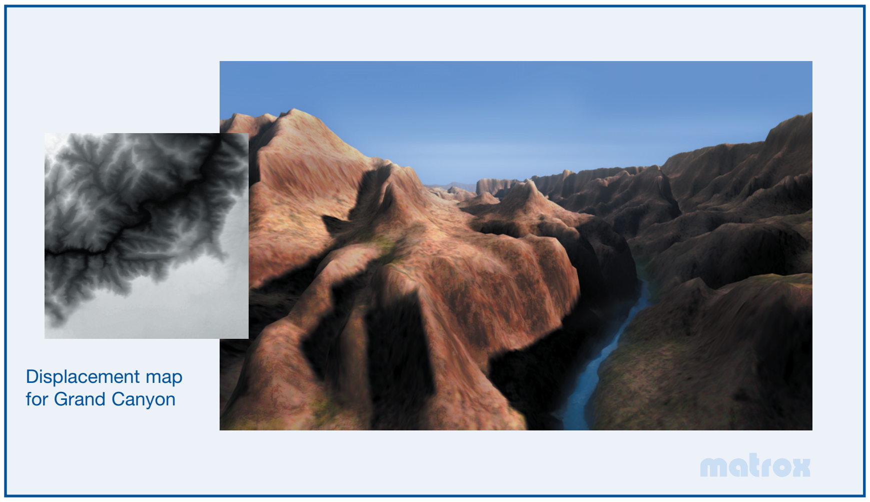


Figure 8
Grand Canyon rendered using Hardware Displacement Mapping.

Compact representation of high detail geometry with Hardware Displacement Mapping

- Hardware Displacement Mapping is one of the best surface descriptors for high-resolution geometry. It provides the most compact and simple way of representing data for the dynamic generation of high detail geometry. With Hardware Displacement Mapping, high-resolution geometrical data is encoded into a compact displacement map. Unlike raw vertex data, this encoded data does not need to include the direction of displacement, connectivity information, colors or texture coordinates—these can be interpolated from the base mesh. Encoding geometry data into displacement maps also reduces the storage requirements on media that hold the models and lessens the burden on system memory and local video memory during runtime. Furthermore, developers can create multiple displacement maps for the same base mesh to produce models that differ significantly from one another (see figure 7).

As seen in figure 3, a 4 KB displacement map (64 x 64 x 8 bit), can produce a detailed terrain from a flat base mesh of just 32 triangles. In fact, high levels of detail can also be obtained from a base mesh of as little as two triangles (see figure 5).

As 3D scenes become more complex, the amount of data required to represent the increased geometry can be quite significant. This data needs to be transferred from system memory to the graphics hardware. Additionally, geometry data competes with other data, such as textures and instructions, for storage and bandwidth. Both system memory bandwidth and AGP bandwidth are limited and can often be a bottleneck for performance. By compressing 3D models into a low detail base mesh and a displacement map, Hardware Displacement Mapping also helps reduce the effect of such bandwidth bottlenecks while actually increasing the amount of detail when an object is rendered.

Hardware Displacement Mapping

Hardware Displacement Mapping versus bump mapping

Displacement mapping, while conceptually sharing certain similarities with bump mapping, is a far superior technique and produces results that cannot be obtained using bump mapping. First implemented in graphics hardware by Matrox in 1999, bump mapping uses a bump map on textures to produce an illusion of depth and detail on the surface of objects. The values of the bump map are used to perturb the fetches into texture and environment maps during texture mapping and environment mapping. Even though bump mapping can produce certain compelling 3D effects, it merely creates an illusion and does not actually generate new geometry (see figure 9). Other bump mapping techniques produce similar results and are, therefore, not comparable to Hardware Displacement Mapping.

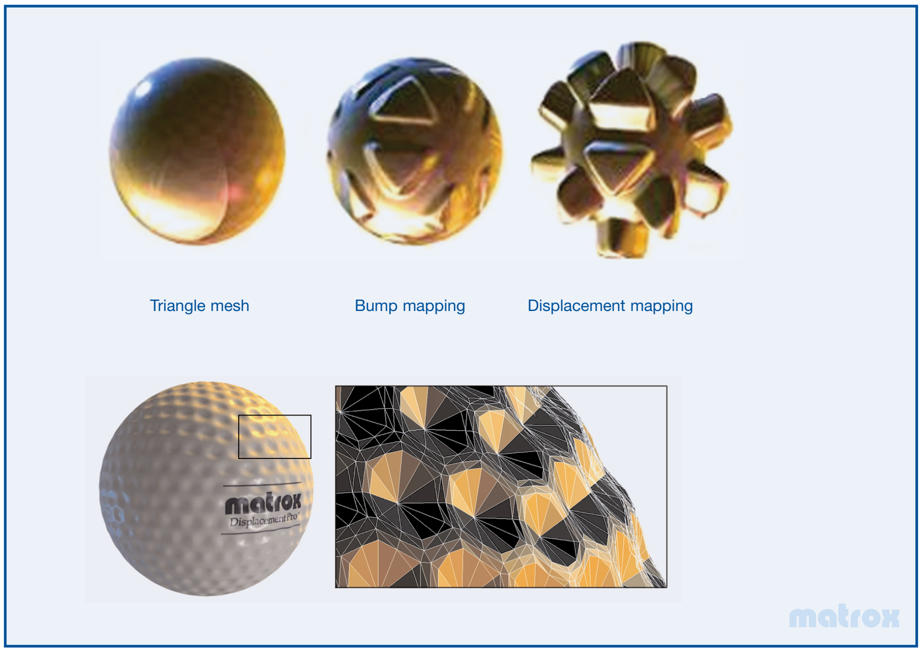


Figure 9
While bump mapping creates the illusion of depth, Hardware Displacement Mapping generates real new geometry.

Hardware Displacement Mapping as an industry standard

As mentioned previously, Microsoft has included Hardware Displacement Mapping (HDM) as an industry standard in its next-generation DirectX API. This helps define a common HDM interface for all developers and facilitates the adoption of this technology.

Additionally, Matrox has developed tools, which enable developers to author their content for Hardware Displacement Mapping. A thorough software development kit (SDK) is also available for qualified software developers to help them implement displacement mapping into their engines. To obtain a copy of the SDK, contact devrel@matrox.com.

matrox
matrox.com/mga

Conclusion

■ Hardware Displacement Mapping (HDM) is an extremely powerful technique that can add an enormous amount of detail to any type of mesh by generating additional geometry. HDM encodes high-resolution data into compact 2D displacement textures, thereby providing a simple and compact representation for high detail geometrical data. The combination of Depth-Adaptive Tessellation and Vertex Texturing provides the most flexible technique for implementing level of detail tessellation in a 3D scene, while maximizing quality, efficiency and performance. Many developers are excited about using this technology to provide the most realistic 3D experience to end users.

Benefits of Hardware Displacement Mapping include:

- Unprecedented detail and realism in 3D scenes with real geometric extrusion
- Simple and compact representation of high-resolution geometry data
- Maximum scalability and efficiency through Depth-Adaptive Tessellation and mip-mapping of displacement maps
- Ideal for a wide variety of 3D meshes including static and dynamic meshes
- Seamless integration into the 3D pipeline along with vertex and pixel shaders
- Defined as an industry standard through future versions of Microsoft's DirectX API