White Paper

DX10FeatureDemo

NVIDIA Confidential
Prepared and Provided Under NDA

February 2007
WP-03017-001_v01

Document Change History

Version	Date	Responsible	Reason for Change
	February 15, 2007	EM, TS	Initial release

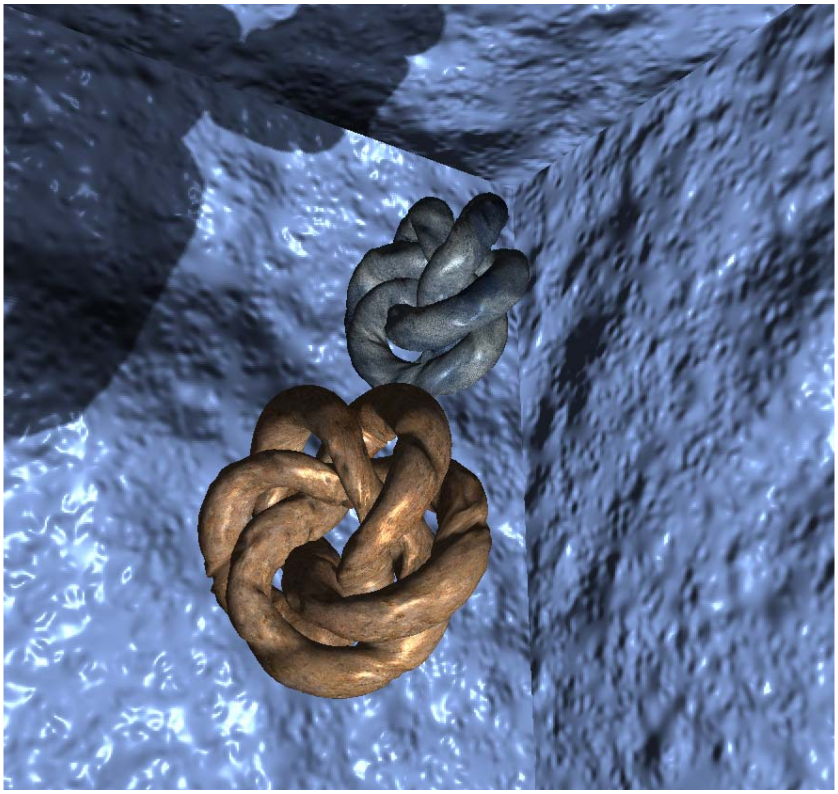
Go to sdkfeedback@nvidia.com to provide feedback on Clipmap.

DX10FeatureDemo

Abstract

DirectX 10 has many new features that can be used for real-time rendering, to simplify existing techniques and introduce new ones. The application described here shows how to use most of the new features in DirectX 10 to efficiently render a simple animated scene. This scene is synthetic and is designed not to show one effect or technique, but to show several features, not emphasizing any one. Though this scene is quite simple and could be rendered using previous APIs, the goal is to show how such a scene can be rendered using DirectX 10, and to provide sample use cases for the major features in DirectX 10.

Andrei Tatarinov
NVIDIA Corporation



Motivation

The application is intended as an illustration of DirectX 10. So it uses as many DirectX 10 features as possible, combined to render a single scene.

It is not suggested that the final visual appearance of the demo would be directly useful in games or other applications; it is very abstract and synthetic. But the techniques, such as cube-map shadow maps, are certainly useful. This sample can be used as an initial DirectX 10 application when embarking on a new project.

How DX10FeatureDemo Works

Instancing is used to render two torus knots, each with its own transformation matrices. The SV_InstanceID is used to select a diffuse texture and a height map from texture arrays.

The torus knots are generated on the fly in the vertex shader. The SV_VertexID is used to calculate parameters for the torus knot equation, which – along with its second derivative – is used to obtain a point on the torus knot surface and a normal to this surface at a given point. Vertex texture fetches from a height map are used to displace the obtained point along its normal.

The surrounding background box is rendered using multi-indexing because different faces which share edges do not share texture coordinates. Two buffers are bound as shader resources – one contains an array of vertices and another contains an array of texture coordinates, and each vertex in the vertex buffer contains a pair of integer values, which are indices into the previously bound buffers. A Load instruction is used to fetch from these buffers.

The dynamic height map on the box faces is generated in the pixel shader in real-time using 3D Perlin noise. A slice of 3D noise is mapped on each box face, using SV_PrimitiveID to find which face we are rendering and to select the corresponding slice. A value obtained by summing a set of noise frequencies is used as a height value for a given pixel. ddx and ddy instructions are used to quickly calculate a normal to the surface at a given point.

Shadowmaps are used to generate shadows on the faces of the box. Shadowmaps are rendered to a cube map in a single pass using the geometry shader. The geometry shader takes a single triangle as input, and returns six triangles as an output. Each output triangle is transformed with an appropriate transformation matrix and assigned to one of the six render targets. Each render target corresponds to one cube map face.

Notice

ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE.

Information furnished is believed to be accurate and reliable. However, NVIDIA Corporation assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use. No license is granted by implication or otherwise under any patent or patent rights of NVIDIA Corporation. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. NVIDIA Corporation products are not authorized for use as critical components in life support devices or systems without express written approval of NVIDIA Corporation.

Trademarks

NVIDIA, the NVIDIA logo, and GeForce are trademarks or registered trademarks of NVIDIA Corporation in the United States and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050
www.nvidia.com