UncertaintyVisualizer

Authors: *Thomas Viard, Guillaume Caumon and Bruno Lévy*

Disclaimer

UncertaintyVisualizer is a stand-alone application developed by the Gocad Research Group; it reproduces some of the features available in the UncertaintyViewer plugin. The source code, executables and other materials attached to the UncertaintyVisualizer package are provided “as-is” and without warranty of any kind, express, implied or otherwise, including without limitation, any warranty of merchantability or fitness for purpose. In no event shall the authors nor the Gocad Research Group be liable for any special, incidental, indirect or consequential damages of any kind, or any damages whatsoever resulting from loss of use, data or profits, whether or not advised of the possibility of damage, and on any theory of liability, arising out of or in connection with the use or performance of this software.

The source code, executables and other materials attached to the UncertaintyVisualizer package can be used and redistributed without explicit permission, as long as the following conditions are met:

- Redistribution of the source code, executables and other materials attached to the UncertaintyVisualizer package must retain the above disclaimer and this list of conditions;
- Redistribution of the source code must retain the mention: “This software contains source code provided by NVIDIA corporation”;
- Publications or derived work using this software must acknowledge the Gocad Research Group and the authors;
- The name of the contributors to this software may not be used to endorse or promote products derived from this software without specific prior written permission.

Requirements

Because UncertaintyVisualizer exploits the programmability of graphic processing units (GPUs), it will only run on programmable graphics cards (i.e. on a GeForce 4 Series or more advanced).

Introduction

UncertaintyVisualizer is a stand-alone application which aims at corendering a petrophysical property (also referred to as the “primary variable”) together with its associated uncertainty, along a voxel-based slice of a geological volume. Two different uncertainty visualization methods are available, which respectively map uncertainty to either texture pattern or blur intensity. Input files are based on an extension of the GSLIB format, described below, so that this software can be used on custom datasets. The source code is provided with documentation, so that it can be adapted for specific needs.

General user commands

The scene can be moved with the mouse: a left click will trigger scene rotation, a middle click will move the whole scene across the view plane, and a right click will change the zoom level.

By default, no uncertainty is displayed. Texture-based uncertainty visualization can be activated by pressing key ‘t’ and blur-based visualization with key ‘b’. In texture mode, the keys ‘+’and ‘-’ can be used to increase or decrease the maximum texture intensity. In blur mode, the keys ‘+’and ‘-’ can be used to increase or decrease the Gaussian blur radius. In both uncertainty visualization modes, the key ‘r’ can be used to reverse the intensity function I(u), which acts as a filter on uncertainty values.

Properties used as the primary variable and the uncertainty can be changed at will, respectively by pressing key ‘v’ and key ‘u’.

Texture-based uncertainty visualization

Our texture-based visualization algorithm maps the primary variable to the background slice color, and associated uncertainty to the intensity of a “fabric” texture pattern.

By default, textured areas correspond to low-uncertainty areas. This behavior can be changed by reversing the intensity function I(u).

If the uncertainty display interferes with the background color-coded primary variable, the maximum texture intensity can be decreased to make the texture pattern less visible.

Blur-based uncertainty visualization

Our blur-based visualization algorithm maps uncertainty to the blending ratio between a sharp and a fully-blurred display of the background color-coded primary variable.

By default, blurred areas correspond to high-uncertainty areas. This behavior can be changed by reversing the intensity function I(u), although it is much more intuitive to associate blur with poorly known areas.

If the uncertainty display interferes with the display of the primary variable, the radius of the Gaussian can be decreased to make blur less distractive.

Note: if the blur radius is increased, the display will strongly slow down due to a large number of texture access and exponential function calls at every frame. In practice, the blurred display should be precomputed once, in order to keep the display interactive. We did not implement this feature to keep the source code of UncertaintyVisualizer as simple as possible.

Input data

UncertaintyVisualizer takes extended GSLIB files as an input for the petrophysical properties and their associated uncertainty, which is assumed to be one scalar value. The GSLIB file "data.gslib" is read when UncertaintyVisualizer; it can be found in the "data" folder. If you want to use UncertaintyVisualizer with your own data, you can overwrite the content of this file. Caution, your custom data should be in the following format:

1. The first line should contain the number of voxels *UMax* along the U axis, followed by the number of voxels *VMax* along the V axis. These two figures should be integers, and must be separated by a space or tabulation. This requirement is the main difference with the standard GSLIB file format.
2. The second line should contain the number of properties *NbProps*.
3. The *NbProps* next lines should contain the names of the properties.
4. The *UMax*VMax* next lines should contain the property values, in the same order as property names were given, separated by spaces or tabulations. The property values should be ordered so that first property values line is *{U=0,V=0}*, second is *{U=0,V=1}*, and so on until *{U=0,V=VMax-1}*. The *VMax*+1th line will then be *{U=1,V=0}*, *etc*.

The "data.gslib" file provides an example of what the expected file format looks like.

A few comments on the file format requirements / UncertaintyVisualizer limitations:

- Even if ratio *UMax*/*VMax* is different from 1, the petrophysical properties will be rendered on a square. This will cause distortions in the display (data should be visually stretched over the larger axis).
- UncertaintyVisualizer currently does not support GSLIB files across volumes, i.e. GSLIB files with more than one voxel along the W axis. If such files are used, the results are unpredictable.
- The property names should not contain any space or tabulation. If they do, only the first part of the name will be kept.
- Up to four properties can be used. All properties after the fourth one will be ignored.
- The property values should be normalized between 0 and 1, because the range of the colormap and texture/blur intensity is expected to be [0,1]. If property values are below 0 or above 1, they will be clamped to either 0 or 1.

Source code design

The source code of UncertaintyVisualizer is divided into two different parts:

(i) Some CPU code that is responsible for setting up the GUI, loading the data and sending appropriate input to the graphics card through the OpenGL API. The CPU code is written in a combination of C and C++.

(ii) Some GPU code that runs directly in parallel on the graphics card, which is responsible for computing the color of every single rendered pixel. This GPU code is written in the OpenGL Shading Language (aka GLSL). This source code is stored in char arrays; it can be compiled and linked at run-time by the pilot of the graphics card.