NVIDIA IndeX

Overview

17 January 2025
Version 2.0

NVIDIA IndeX – Overview

Document build number rev384410

Contents

1 Background

The real-time visualization of *volumetric data* is essential for experts in a variety of fields to achieve visual insight. Medical, meteorological and geological software applications transform measurements into images that enable understanding previously unavailable to the analyst. Volumetric data displayed by these applications can be produced by both simulation and measurement. For example, weather simulations model the laws of atmospheric dynamics; magnetic resonance imaging provides important diagnostic information about structures in the human body. The amount of data produced during simulation and measurement, however, can be extremely large, and can challenge traditional visualization methods. In particular, acquisition techniques applied in the oil and gas industries that scan the Earth's subsurface structure generate terabytes and even petabytes of *n*-dimensional volumetric data that cannot be visualized using individual workstations alone.

2 The solution provided by NVIDIA IndeX

NVIDIA IndeX is a commercial software solution that leverages *GPU clusters* for real-time scalable visualization and computing of multi-valued volumetric data together with embedded geometry data.

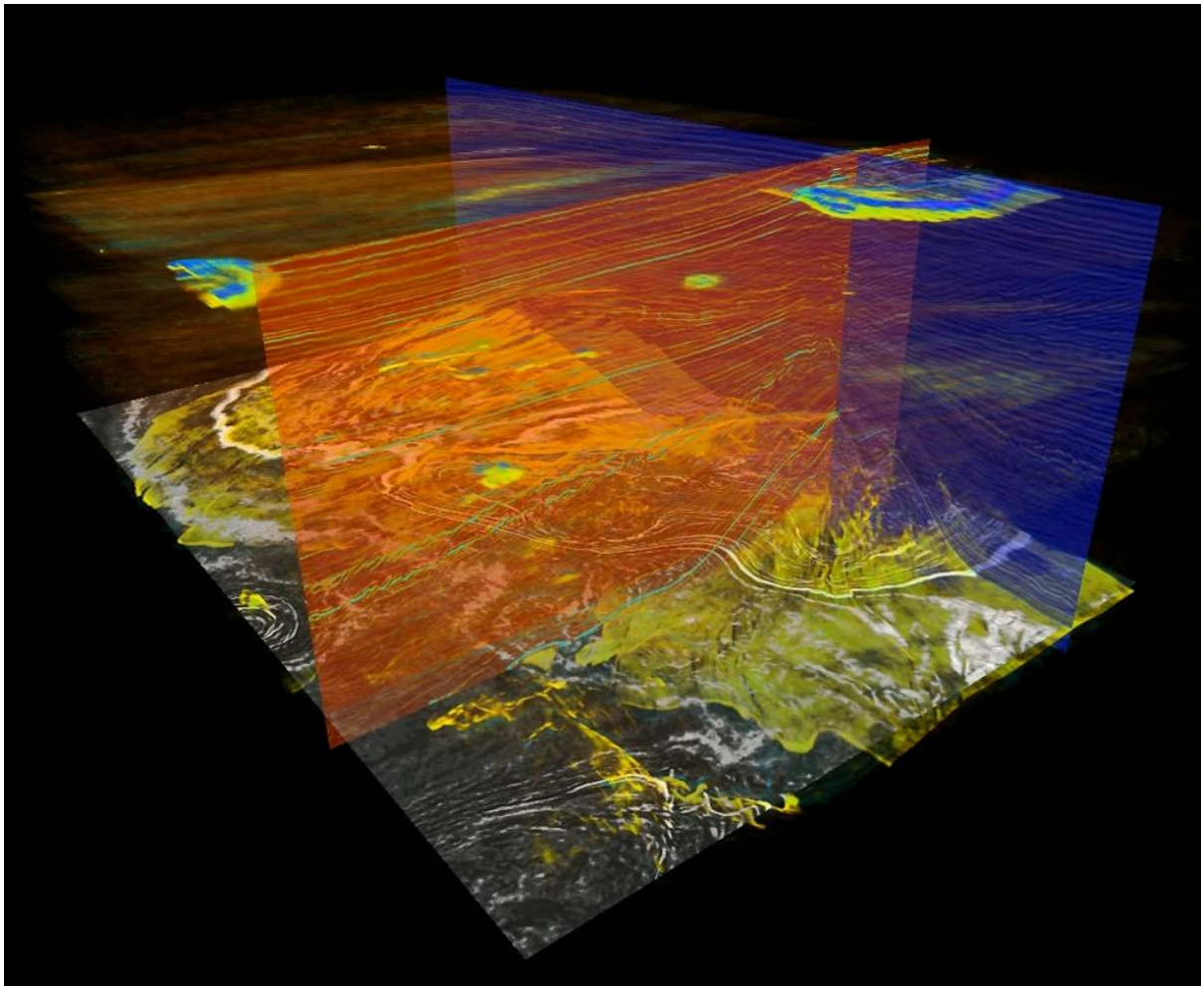

Fig. 2.1 - Volume data and slices through the data rendered by NVIDIA IndeX

The enormous growth in dataset sizes — especially in the oil and gas industries — requires software solutions that can scale with GPU cluster sizes to efficiently handle and visualize the large datasets produced today as well as the larger datasets of the future. NVIDIA IndeX overcomes the limitations of a single computer system (for example, the CPU/GPU bandwidth constraints) by managing the shared resources of a set of computers treated as a cluster, each computer equipped with one or more GPUs.

3 The main features of NVIDIA IndeX

NVIDIA IndeX addresses six key areas to satisfy the requirements of data visualization for use in analysis by domain experts:

- *Visualization* — Accurate, high-quality data visualization, feature representation and annotation
- *Scalability* — Effective data rendering and display at any resolution
- *Scene management* — High-level scene description and management
- *Interaction* — Infrastructure for the creation of rich visualization interaction techniques
- *GPU computing* — High-performance volume visualization and analysis using the GPU
- *Extensibility* — Flexibility and extensibility for a variety of application domains

The implementation of these features is supported by the three primary software components of the NVIDIA IndeX architecture:

- The visualization layer
- The data distribution and compute layer
- The extended features layer, enabling:
  - Video streaming-based applications
  - Client-server deployments, including thin-client implementations
  - Multi-user interaction

3.1 Visualization

The visualization component of NVIDIA IndeX handles the distributed, parallel rendering of large amounts of *n*-dimensional data. This component implements two phases of real-time image generation:

- *Rendering* — The GPU-based cluster-wide high-quality and depth-correct real-time visualization of semi-transparent volumetric data with embedded geometry using dedicated ray-casting and ray-tracing techniques based on NVIDIA CUDA.
- *Compositing* — The cluster-wide combination of intermediate rendering results for final image synthesis.

In addition, the visualization component facilitates the accurate, high visual quality rendering of original data.

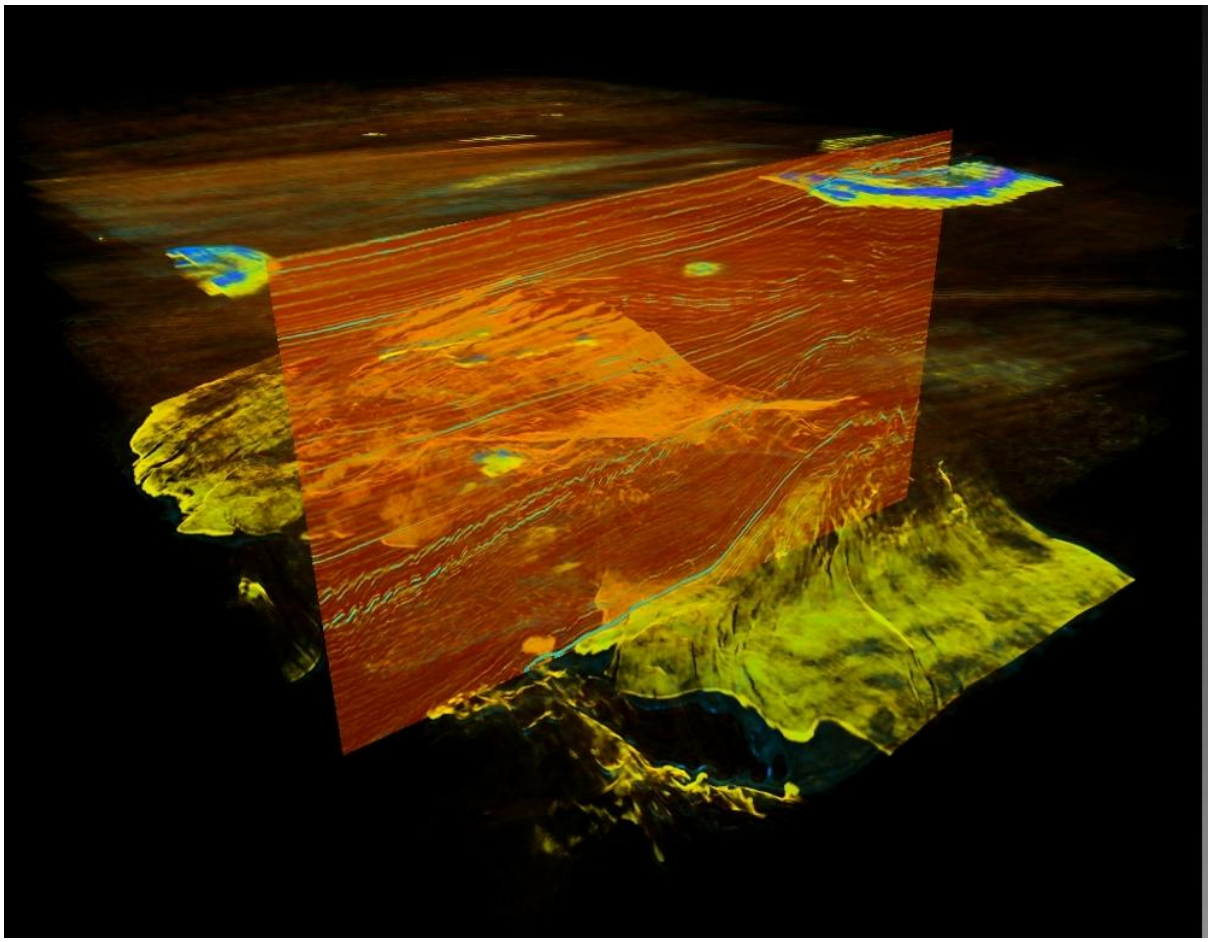

Fig. 3.1 - A variation on the visualization parameters of Figure 3.1.

3.1.1 Accuracy

Specialized ray-casting, ray-tracing and rasterization algorithms have been designed for NVIDIA IndeX. The combination of the different rendering algorithms have been implemented in such a way that the accurate depth-correct rendering of the opacity or semi-transparency of the various types of shapes (such as triangle geometry, volumes, points and lines) embedded in a scene is possible. For instance, the renderings in Figure 3.2 show a horizon dataset displayed within a seismic volume. The opaque horizon occludes the seismic data below the horizon's surface while the seismic data becomes visible if the horizon is semi-transparent.

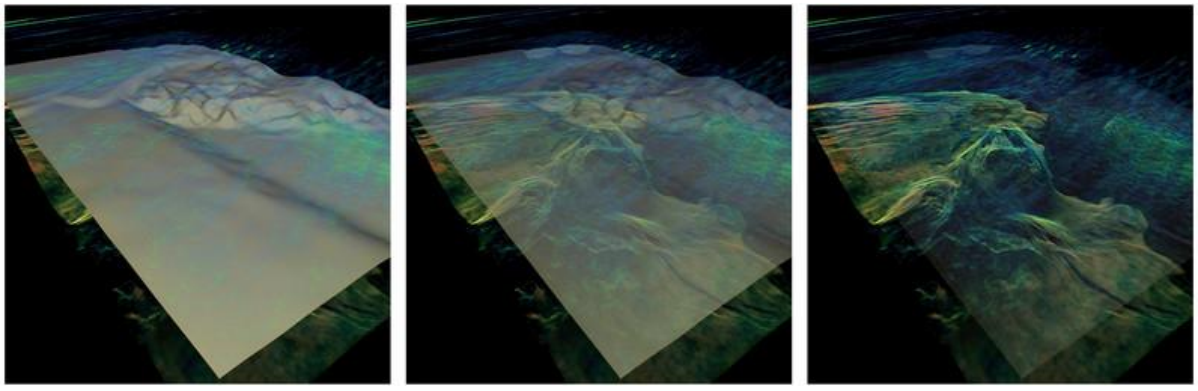

Fig. 3.2 - Volume data below transparent polygonal meshes are displayed correctly with respect to depth.

The accuracy and correctness of the data visualization offered by NVIDIA IndeX facilitates the visual assessment of data and their relationship, which highly accelerates decision-making processes by analysts.

The increasing importance of accurate visualization for reservoir engineering is addressed by NVIDIA IndeX by its direct support of reservoir grids. In Figure 3.3, a synthetically generated reservoir data set rendered by IndeX shows its capabilities for fine detail rendering from cellular geometric data.

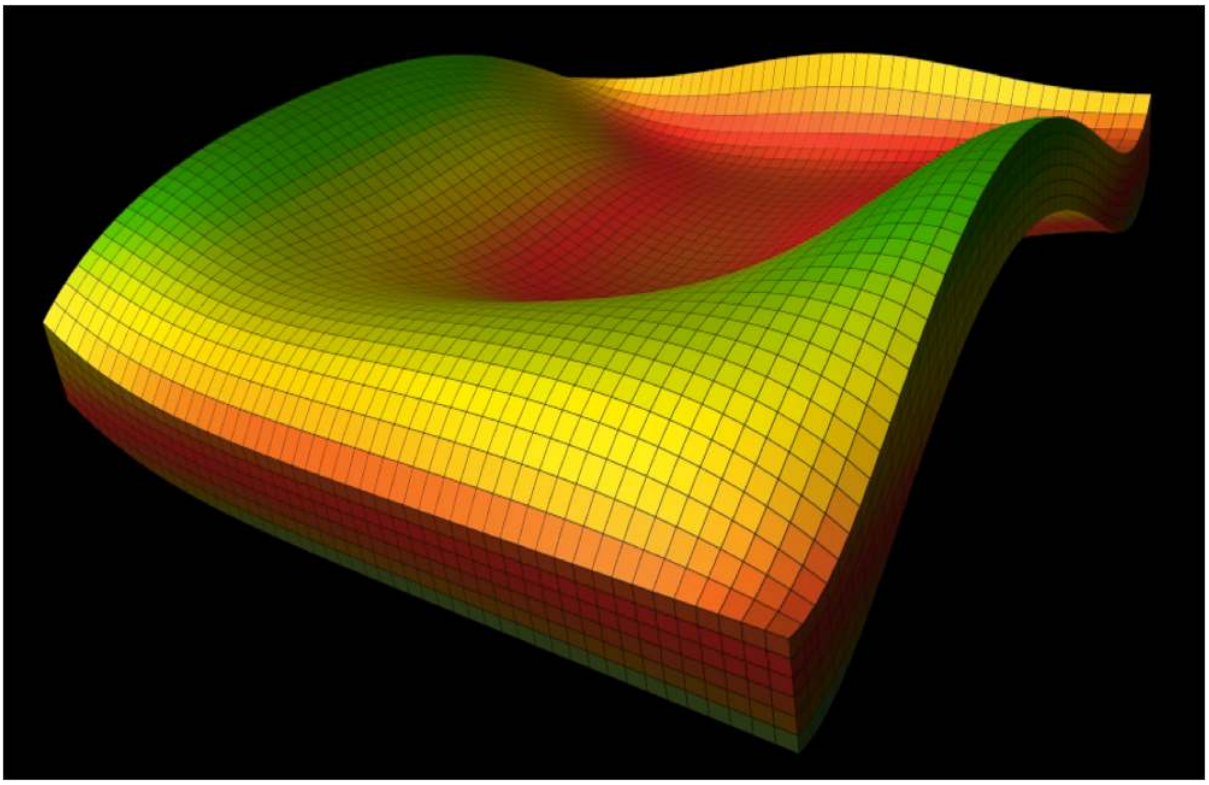

Fig. 3.3 - Synthetic reservoir dataset rendered by NVIDIA IndeX

NVIDIA IndeX also accurately combines different data modalities for the unique insight that simultaneous visualization can provide, as shown in Figure 3.4 (page 6).

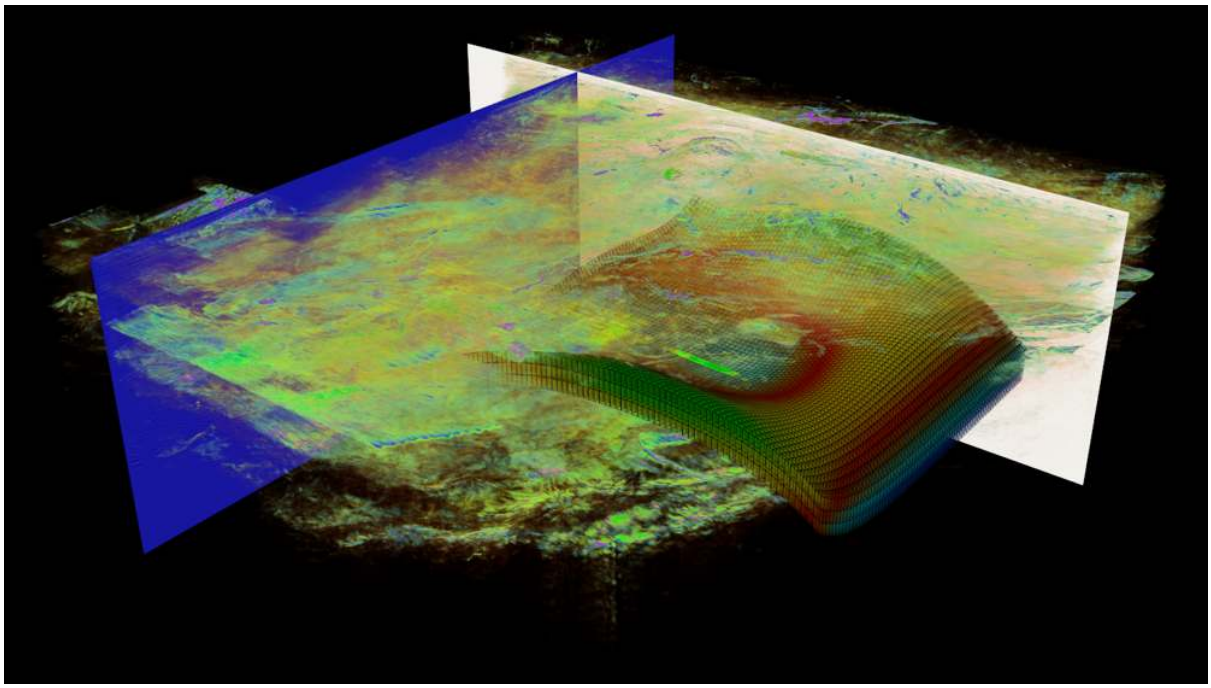

Fig. 3.4 - Depth-correct rendering of seismic data — volumes, heightfields, slices — combined with a visualization of reservoir data

3.1.2 Visual quality

Besides correctness and accuracy, NVIDIA IndeX implements techniques that deliver images of high visual quality from the data. Super-sampling, anti-aliasing, and filtering techniques generate presentation ready quality renderings. Figure 3.5 shows two seismic volumes. The visualizations reveal the inherent fine details of the data without introducing distracting sampling artifacts.

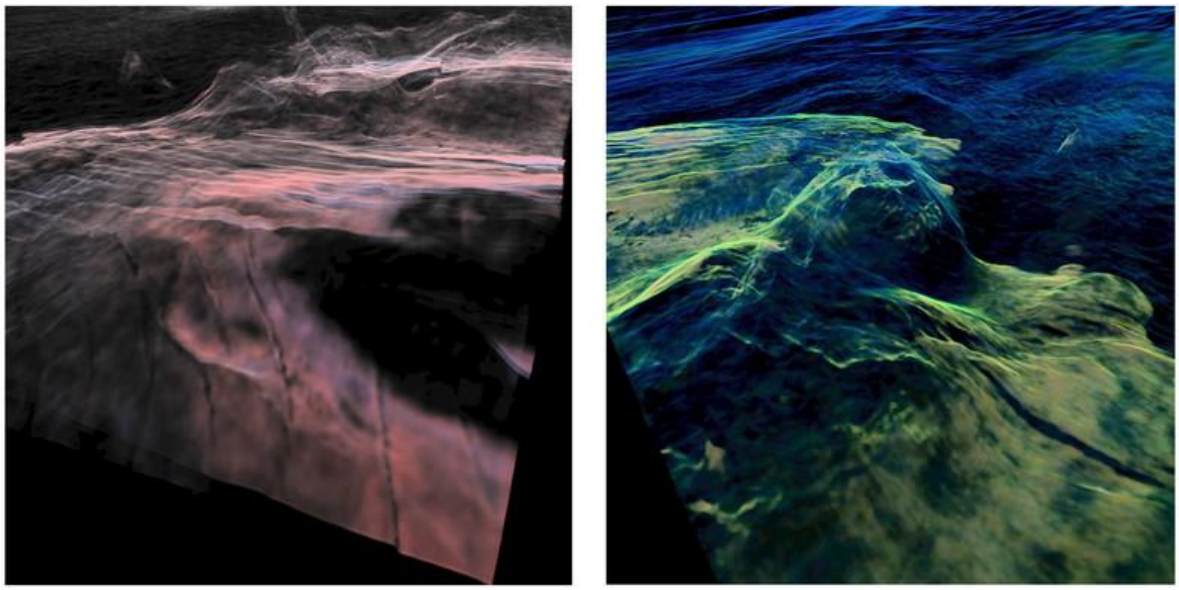

Fig. 3.5 - Rendering of two seismic volumes by NVIDIA IndeX displaying fine detail

Renderings of high visual quality are a vital requirement for data interpreters and analysts given the fundamental role these renderings play in their daily work.

Considerations of visual quality are also important in the manner in which annotational data is represented. For example, Figure 3.6 shows the way in which geometric mesh objects can be customized in their display to best serve the interpreter's visual requirements.

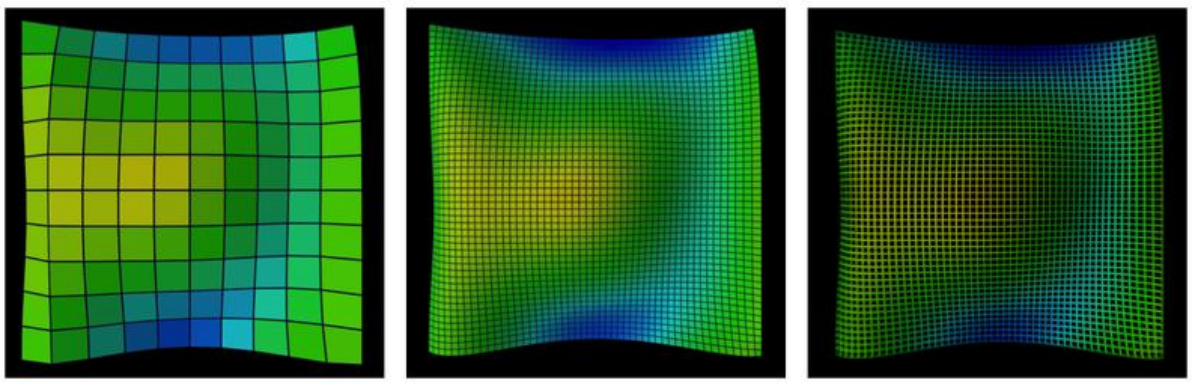

Fig. 3.6 - Mesh outlines, with variations of style, thickness and color

The reservoir grid visualization fully supports semi-transparent cells, and therefore assists in the spatial comprehension of complex datasets. Individual cell properties such as color and opacity be changed interactively, another important tool for the analyst during visualization shown in Figure 3.7.

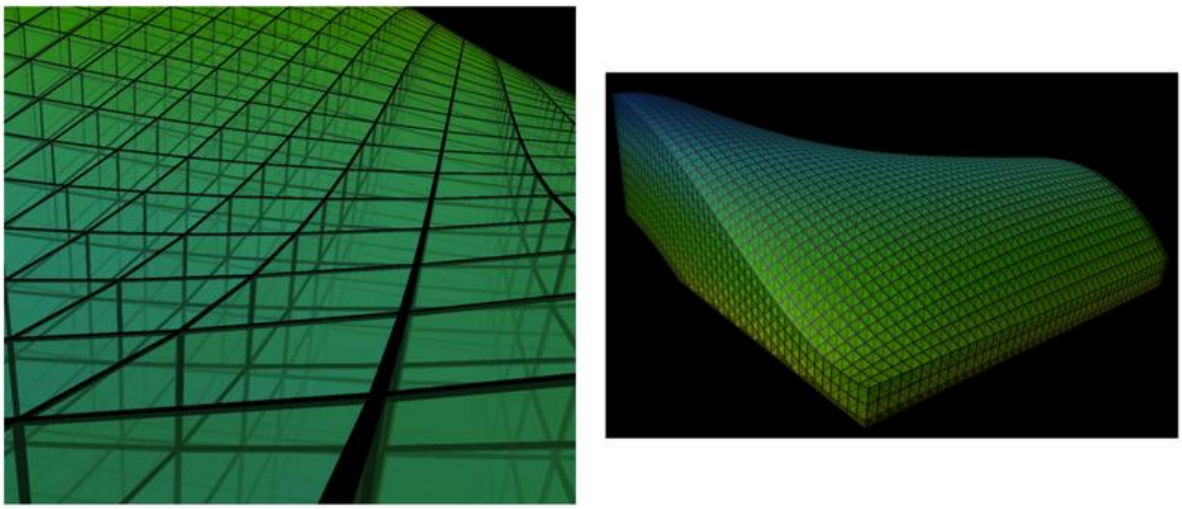

Fig. 3.7 - Transparency of cells in reservoir grid simulation

3.1.3 Original data

At all times, NVIDIA IndeX renders the uncompressed, original data. Level-of-detail approaches that approximate the data were intentionally disregarded in the design of NVIDIA IndeX as being insufficient for its usability requirements. Whereas approximations reduce the visual correctness and cause distracting artifacts when interacting with the data, the solution chosen for NVIDIA IndeX always allows experts like seismic data interpreters to trust the depicted data during their visual analysis.

The ability to base the visualization on the original data only represents a key feature of NVIDIA IndeX and is achieved with the unique, scalable, cluster-based solution.

3.2 Scalability

Central to the design of NVIDIA IndeX is its ability to accurately render very large datasets. However, the detail inherent in such large amounts of data requires the corresponding display of high-resolution imagery for accurate analysis and interpretation. Therefore, the issue of scalability is important for processing as well as for display.

3.2.1 Large data

The parallel, cluster-rendering solution of NVIDIA IndeX partitions the large-scale datasets into *subregions* (Figure 3.8) manageable on a single cluster node (potentially containing multiple GPUs). One cluster node is typically assigned a number of such subregions, where a single subregion is guaranteed to fit into the memory of a single GPU. Usually all subregions of a cluster node can be distributed among its GPUs and stored completely in GPU memory for the actual rendering. Algorithms unique to NVIDIA IndeX are used when all data required for rendering cannot be stored in-core.

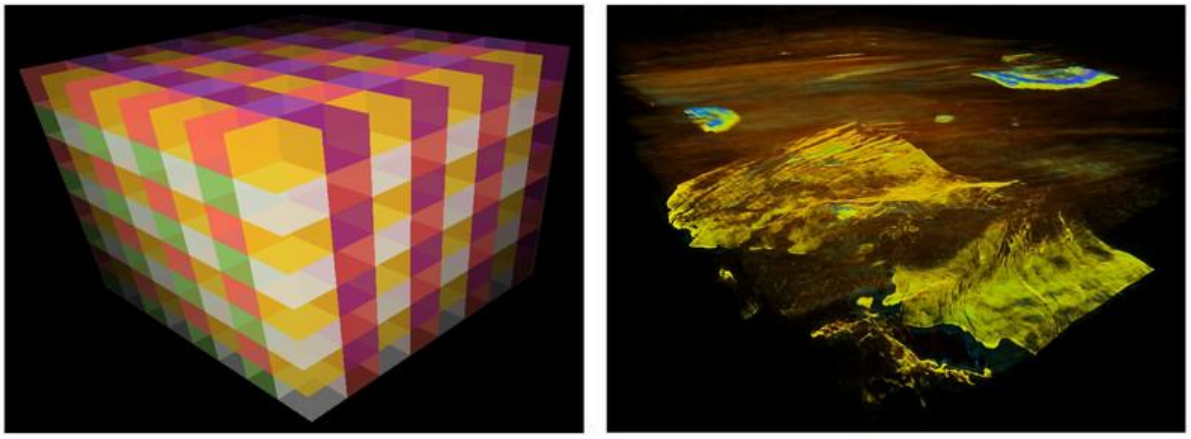

Fig. 3.8 - Division of the scene space into subregions for rendering, with colors corresponding to the host assigned to that subregion

NVIDIA IndeX currently uses a one-stage streaming *out-of-core rendering* approach without *level-of-detail algorithms* and thereby always visualizes the original data and not an approximation of it. The data assigned to a cluster node currently has to fit in its local main memory. However, this data does not need to fit into GPU memory. When encountering such a case, NVIDIA IndeX streams unavailable data into GPU memory by *staging* the rendering process, overlapping rendering with the loading of subregions to be processed and allowing the rendering process to run uninterrupted.

NVIDIA IndeX can display approximately 700GB of semi-transparent three-dimensional volume data at more than 20 frames per second on a GPU cluster consisting of 30 machines, each equipped with four NVIDIA Tesla K10 GPUs. Adding more GPUs to the cluster by extending the number of machines or by extending the number of GPUs per machine further increases the rendering performance. Adding GPUs to the cluster also allows for larger dataset sizes while maintaining rendering performance. The network traffic and the compositing load are mostly hidden by the visualization layer; the difference between finishing rendering on the cluster nodes and compositing the final image is at most 20ms.

In NVIDIA IndeX, the component utilized for cluster-wide, in-memory large data management, for scheduling and distributing compute tasks, and for abstracting from the network

infrastructure is the *Distributed Computing Environment* (DiCE)[1] , a platform for writing parallelized high-performance computing applications. DiCE helps developers write applications that scale to thousands of multi-core CPUs combined with thousands of GPUs. DiCE addresses both the scaling to the CPUs and GPUs of a single machine as well as the optimal use of all the resources in a cluster of machines.

3.2.2 Large display

In all visualization application domains, analysts need high-quality imagery for the correct interpretation of data. Since the quality of standard definition (SD) video can be too poor for accurate interpretation, the resolution of typical implementations is no longer acceptable in many data visualization scenarios. Efficient compositing techniques implemented by NVIDIA IndeX meet these needs by supporting any pixel resolution up to 4096x2160, such as High-Definition (1920x1080) and Ultra HD (3840x2160).

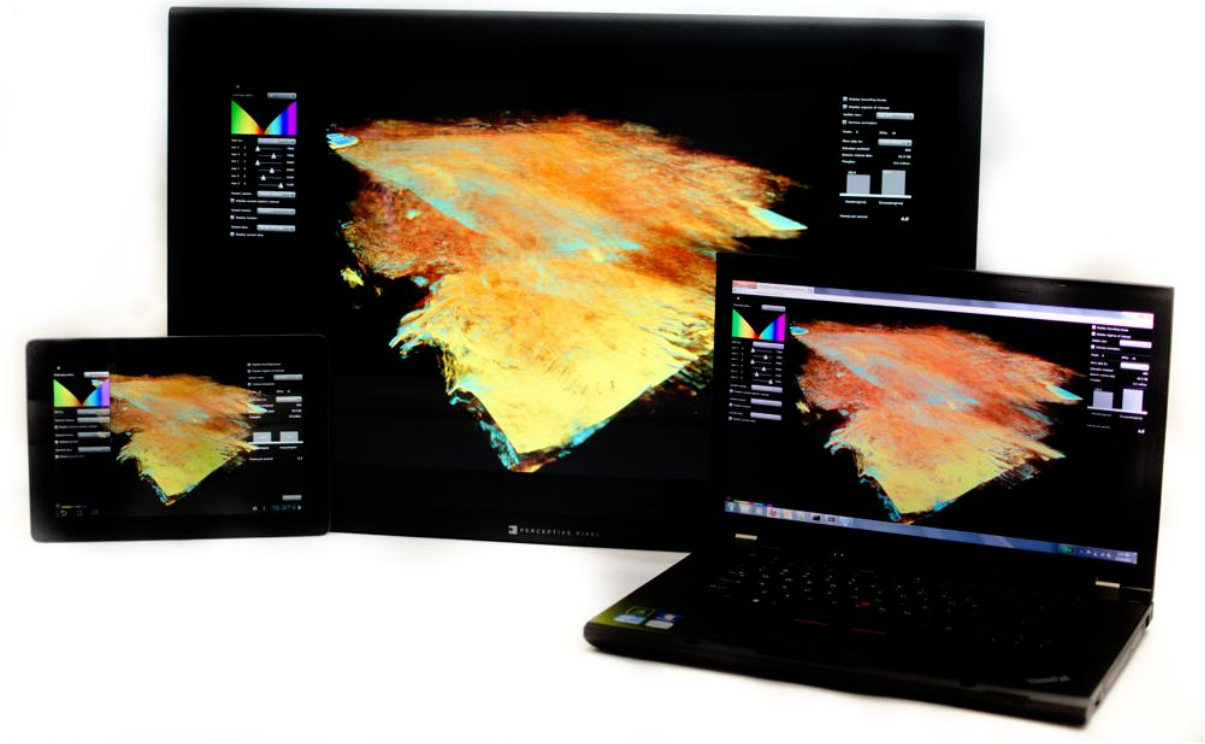

Fig. 3.9 - NVIDIA IndeX reference viewer on a tablet, a laptop, and a high-resolution touch-screen display

For devices that display high-resolution renderings of massive datasets, trade-offs must typically be made to favor either display or data size in the design of the parallel rendering architecture. NVIDIA IndeX implements a hybrid approach, enabling both large-scale data rendering and large-scale display support into a unified system.

During rendering, NVIDIA IndeX divides the screen space into a number of *image tiles*. Each of these image tiles is assigned to a group of machines that form a *subcluster*. These subclusters are then responsible for rendering and compositing the individual image tiles they receive. With this technique it is possible to reduce the network bandwidth requirements when compositing the intermediate results from different machines, thus allowing the display resolutions to

1. *Distributed Computing Environment 3.0 — Functional Overview*, NVIDIA Advanced Rendering Center, May 2012.

reach very large values, such as 4K and higher. The user has the ability to define the subcluster configuration. This configuration specifies the responsibilities of individual machines in a subcluster or of the subcluster as a whole. A machine or subcluster may be responsible for rendering or compositing, or for both processes. However, NVIDIA IndeX also performs automatic subclustering if chosen by the user, thereby ensuring optimal performance without user configuration.

3.3 Scene management

The scene that is visualized by NVIDIA IndeX may contain a multitude of different elements, from simple geometric shapes such as spheres, to large-scale volumetric data or polygonal meshes. These *scene elements* can be modified with attributes and are structured into *scene groups*. Even though scene elements may differ significantly in complexity and rendering costs, they are all handled in a uniform way through a hierarchical scene description. This uniform description allows for the simple and efficient creation of complex scenes and makes powerful interaction techniques possible.

All visible objects in the scene are implemented by two types:

- Large-scale data, represented by volumes, complex polygonal meshes and point clouds
- Geometric shapes defined in three-dimensions (spheres, ellipsoids, cylinders, planes, labels) and in screen space (lines, points, polygons, labels, icons)

Subdividing the scene into several scene groups simplifies the handling of complex scene definitions by allowing logical grouping of related shapes and attributes. The scene is represented by a tree structure where each scene group contains a list of children. The children may be shapes, attributes or other scene groups. The same scene element may be added multiple times at different positions in the tree. This simplifies the reuse of common shapes and attributes, and also reduces resource utilization.

An attribute defines, for example, the material, light or text font that should be used for rendering a shape. A single attribute can be applied to several shapes or even to an entire scene group. Changes in the attribute will have an immediate effect on the rendering of all associated shapes. The assignment of attributes to shapes is handled by the hierarchical scene description, so that scene elements in the scene tree inherit the active attributes from their parent scene groups.

User-defined shapes are supported by a special type of scene group, called a *shape group*. An application using NVIDIA IndeX may implement its own subclass of a shape group and define a new shape by combining existing shapes and attributes. For example, a new arrow shape would be constructed out of a cone and a cylinder. A user-defined method for setting the length of the arrow would ensure that the size of the cone at the head of the arrow would always retain its original size, which would not be possible when just applying a scaling transformation. Once defined, instances of the new shape group can be added to the scene just like predefined shapes.

Scene elements representing large-scale data automatically enforce the rules required by the distributed rendering approach. To prevent costly re-loading from data storage, the object transformation can only be changed when initially adding the element to the scene. For all other scene elements, the transformation can be changed at any time.

The high-level scene description is converted into an optimized low-level representation that allows for efficient rendering. By tracking user changes to the scene description, only the

modified parts of the low-level representation are updated, minimizing the overhead for scene modifications.

3.4 Interaction

NVIDIA IndeX provides full-scale interactive data visualization, but without approximations that modify the data — an essential requirement for analysts. Other solutions that apply *out-of-core algorithms* and *multi-resolution models* that result in a progressive rendering are less acceptable to domain experts who analyze and interpret data; they feel distracted by the inevitable side-effects of those techniques when interacting with the visual display of the data.

In contrast, by using GPU compute capabilities together with cluster compute technology, NVIDIA IndeX can visualize a complete dataset — without approximation — at interactive frame rates. The user can interact not only with volume data, but also with height-field data as well as other geometric primitives, such as lines, triangles, and point clouds.

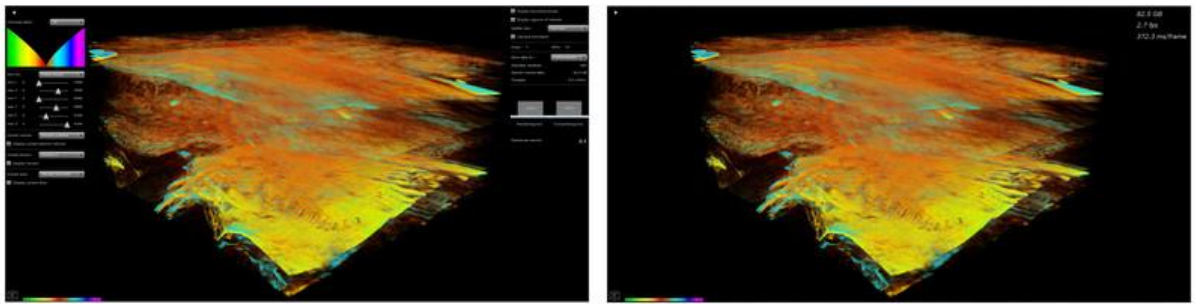

Fig. 3.10 - NVIDIA IndeX reference viewer application in editing and parameter selection mode (left) and fullscreen mode for navigation and analysis (right)

User interaction for applications built with NVIDIA IndeX can be classified into three types:

- *Navigation* — A camera model allows for changing the viewpoint of the visualization through user interface controls and gestures.
- *Editing* — Additional geometric structures, such as isosurfaces represented by polygonal meshes, can be derived from the data and combined with the visualization.
- *Annotation* — Two- and three-dimensional labels and a variety of geometric shapes can be added to the visualization to mark features in the data and add improve legibility.

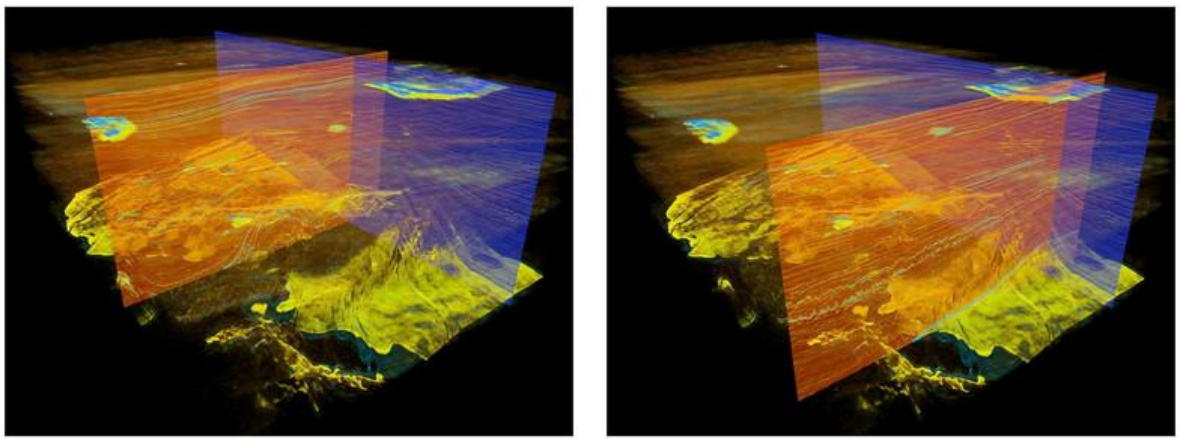

Fig. 3.11 - Defining different slice positions (“roaming”) through a dataset in NVIDIA IndeX

3.5 GPU computing

The large-scale datasets that NVIDIA IndeX visualizes typically result from preprocessing raw data, which can easily be many times larger that the resulting dataset. For instance, it is very common that the three-dimensional attributes of a seismic survey are computed by averaging terabytes of *n*dimensional source data. Parameterizing the averaging process interactively, together with immediate visual feedback of the results, can greatly accelerate the understanding of the data.

NVIDIA IndeX provides the software infrastructure for implementing scalable computing algorithms that possibly run on a separate and dedicated GPU compute cluster for immediate display. Essentially, NVIDIA IndeX enables the ability to bring together compute cycles and rendering cycles in a single interactive system. Being able to leverage the compute power of a dedicated GPU cluster by means of a GPU rendering cluster (Figure 3.12) is game-changing in interactive visual computing.

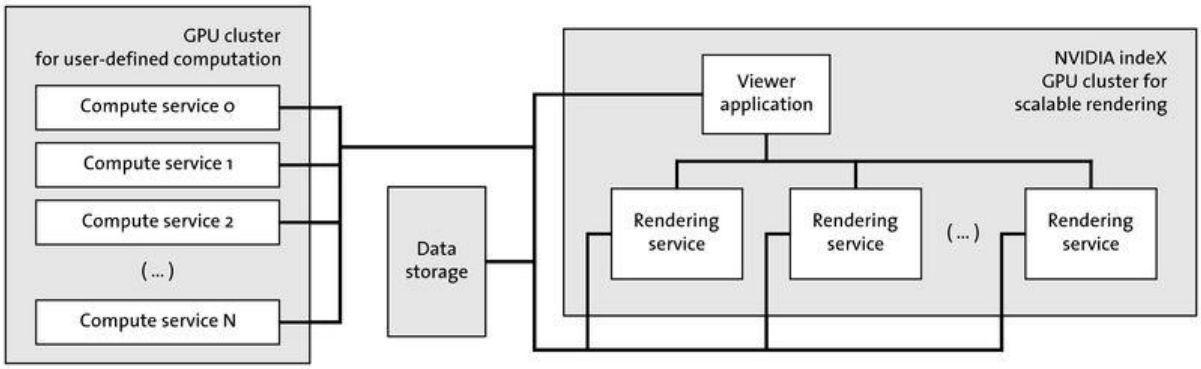

Fig. 3.12 - Leveraging GPU compute and rendering clusters for high-performance visual computing

In practice, NVIDIA IndeX exposes proxy geometries that serve as visualization media. The distributed, parallel rendering of the scene launches compute interfaces for each of the subregions in which the proxy geometry is contained. User-defined implementations of the compute interfaces can then direct their compute tasks to one or many cluster machines. The compute operation is asynchronous, that is, the rendering continues until the compute tasks return. NVIDIA IndeX then integrates the compute results into the rendering by mapping them onto the proxy geometry and displaying them. The combination of the parallel, distributed rendering solution with the asynchronous approach hides the time it takes to generate the results and enable interactive compute for instantaneous data visualization.

3.6 Extensibility

The NVIDIA IndeX extensible software architecture allows programmers to augment an application beyond the native capabilities NVIDIA IndeX provides. The sophisticated C++ API eases the integration of the software in customer products and — together with its optional extended features — supports the customer in implementing deployment strategies.

NVIDIA IndeX can be extended in areas such as custom data importers and exporters that rely on proprietary formats. These formats may support, for example, parallel storage devices. Other important extension areas that are often critical to user-specific domain functionality include:

- Access and manipulation schemes that operate on the data distributed in the cluster

- Analysis methods that derive information from the distributed data
- Compute techniques that either run directly on the data stored on the GPU rendering cluster or even leverage a separate GPU compute cluster

An application programmer extends NVIDIA IndeX using the same programming mechanisms that NVIDIA IndeX itself uses to provide some features, such as filters and the generation of height maps from volume data. The source code for these extensions is provided with NVIDIA IndeX and can be used by the programmer as working examples in developing extensions for an application.

Besides providing the ability to implement application functionality that addresses customer workflows and requirements, this distinct separation between the NVIDIA IndeX solution and customer extensions simplifies the protection of the customer's intellectual property.

For more information about NVIDIA IndeX and the unique capabilities it offers to the application programmer in creating powerful visualization tools, contact the NVIDIA Advanced Rendering Center at `arc-office@nvidia.com`.

4 Image gallery

The cover image and the following pages show the visualization possibilities from a real-world dataset using NVIDIA IndeX. Special thanks to Crown Minerals and the New Zealand Ministry of Economic Development for allowing NVIDIA to display this Taranaki Basin dataset. Crown Minerals manages the New Zealand Government's oil, gas, mineral and coal resources. More information is available from `http://www.crownminerals.govt.nz`.

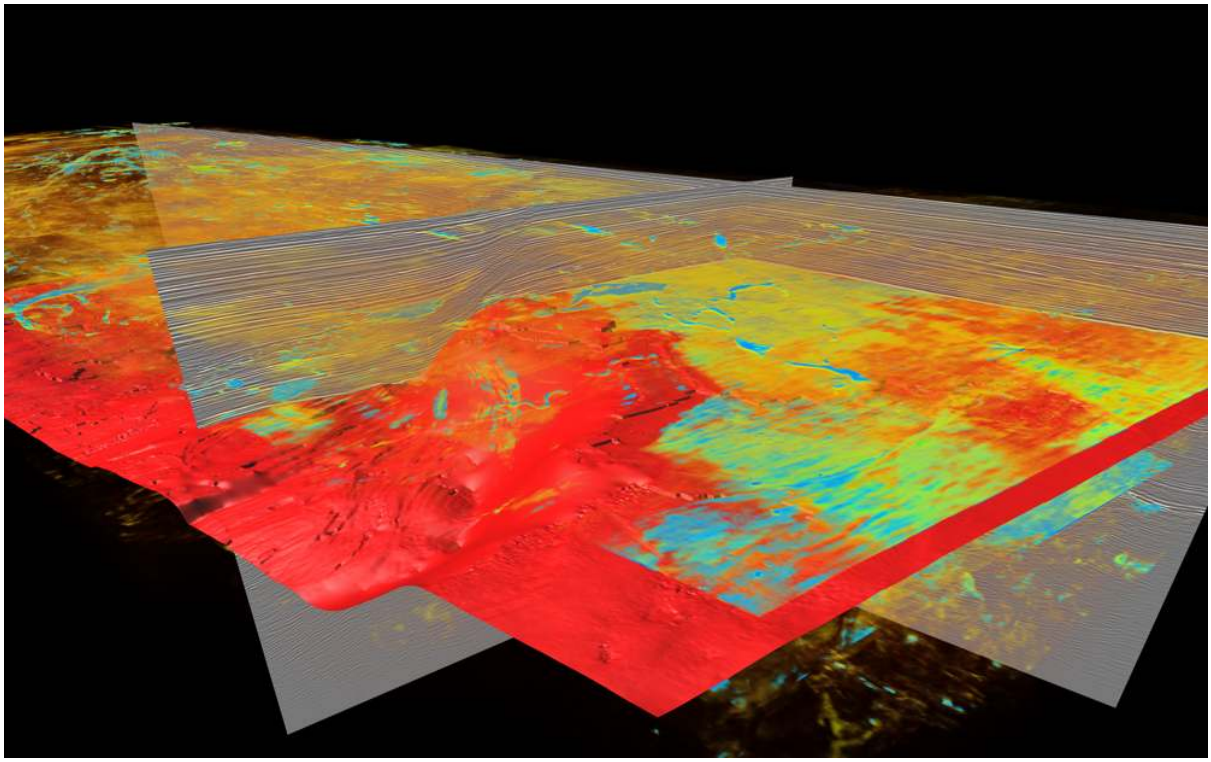

Figure 4.1

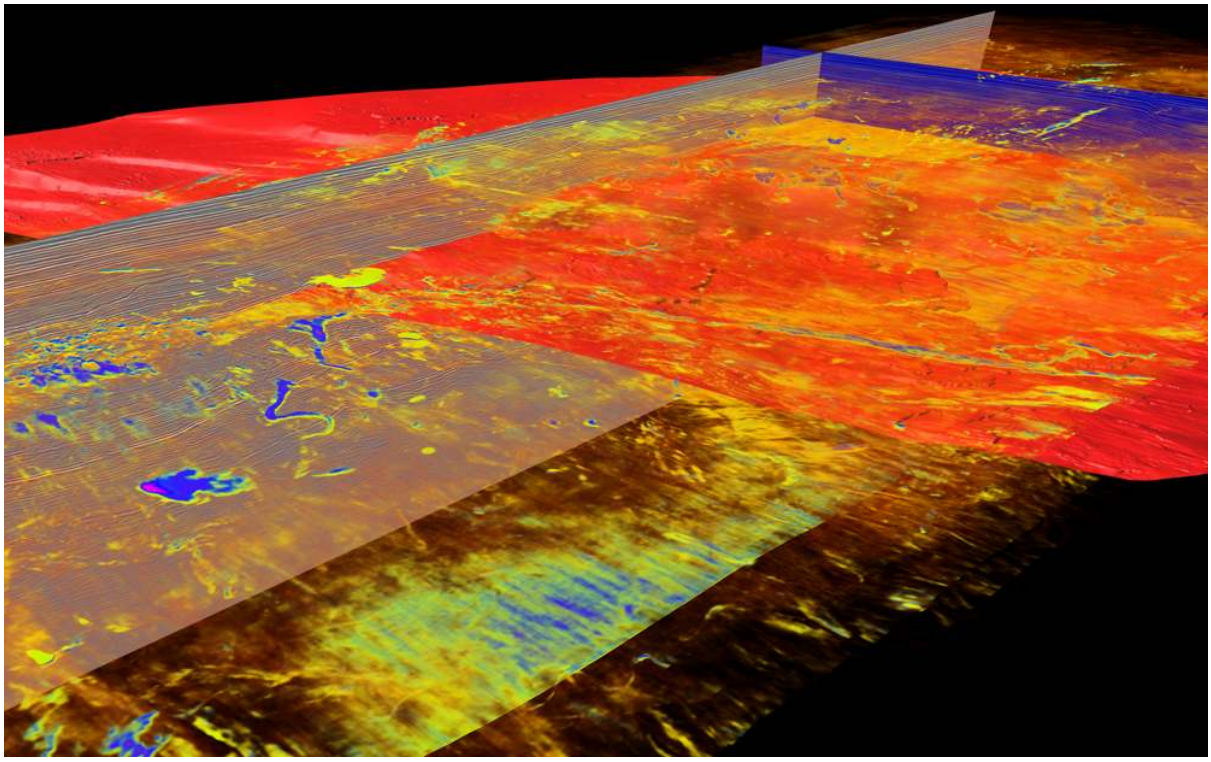

Figure 4.2

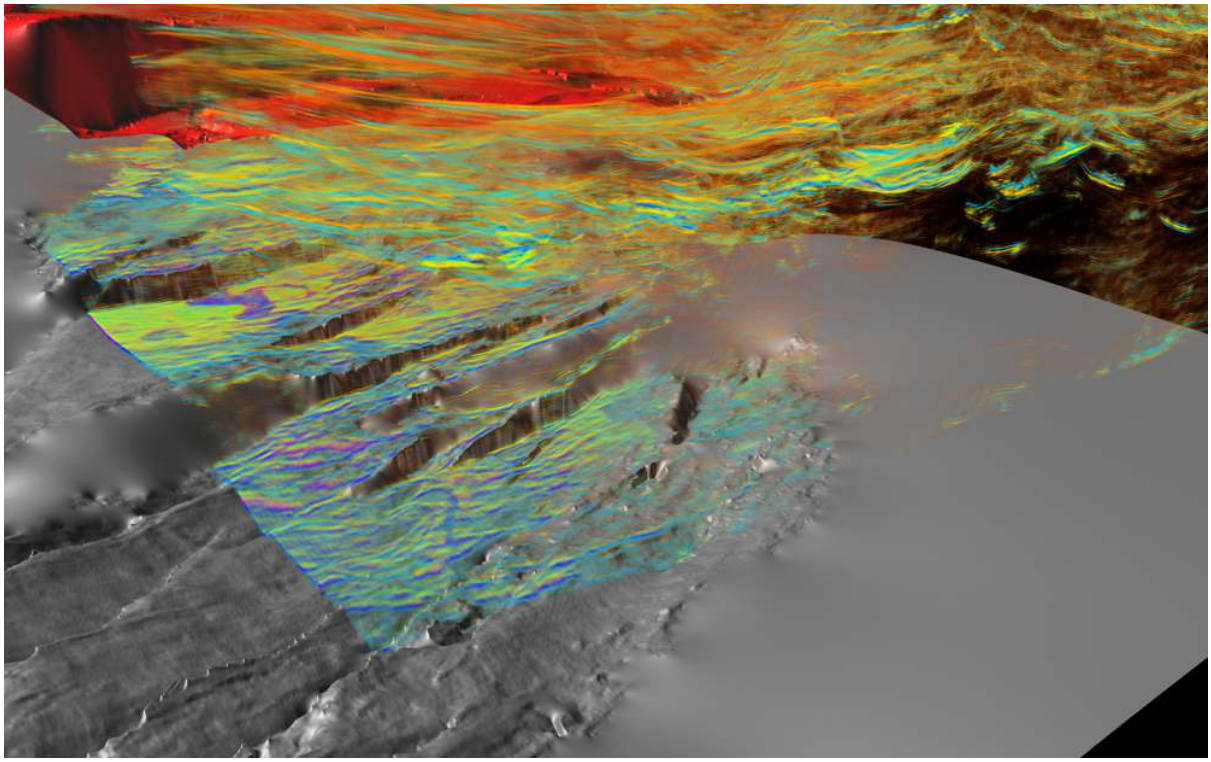

Figure 4.3

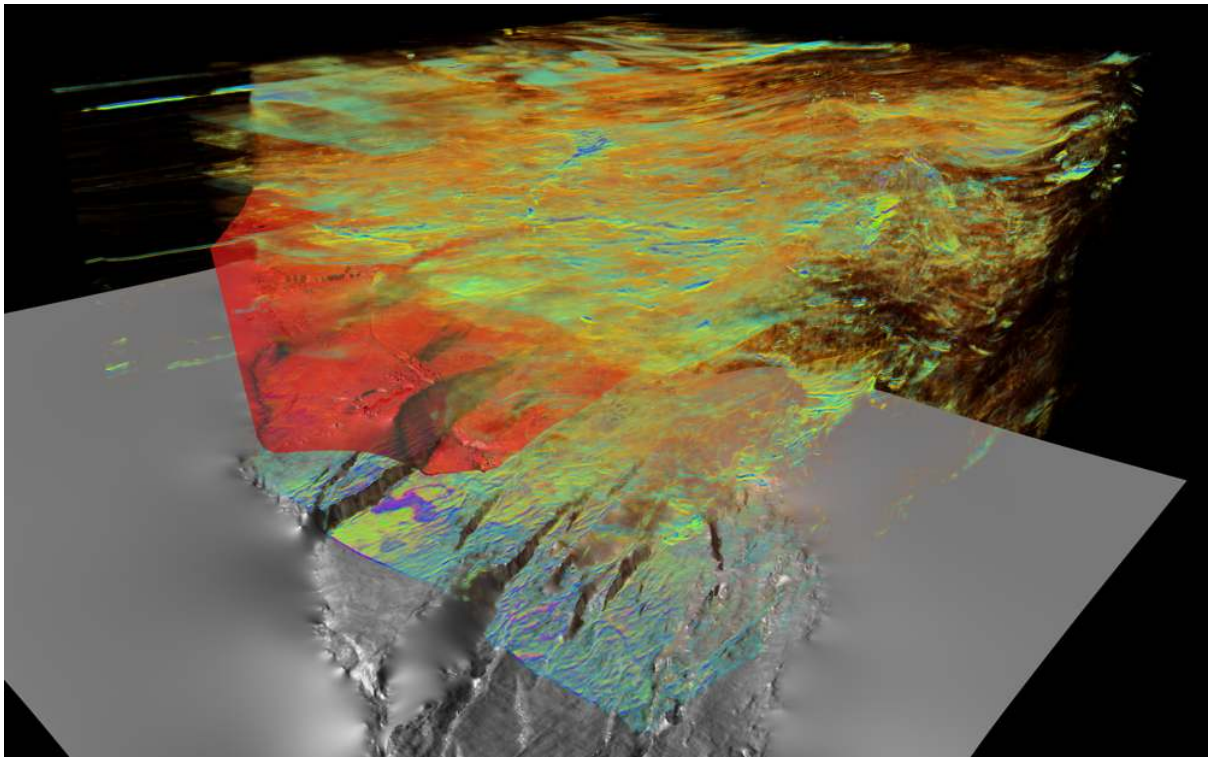

Figure 4.4

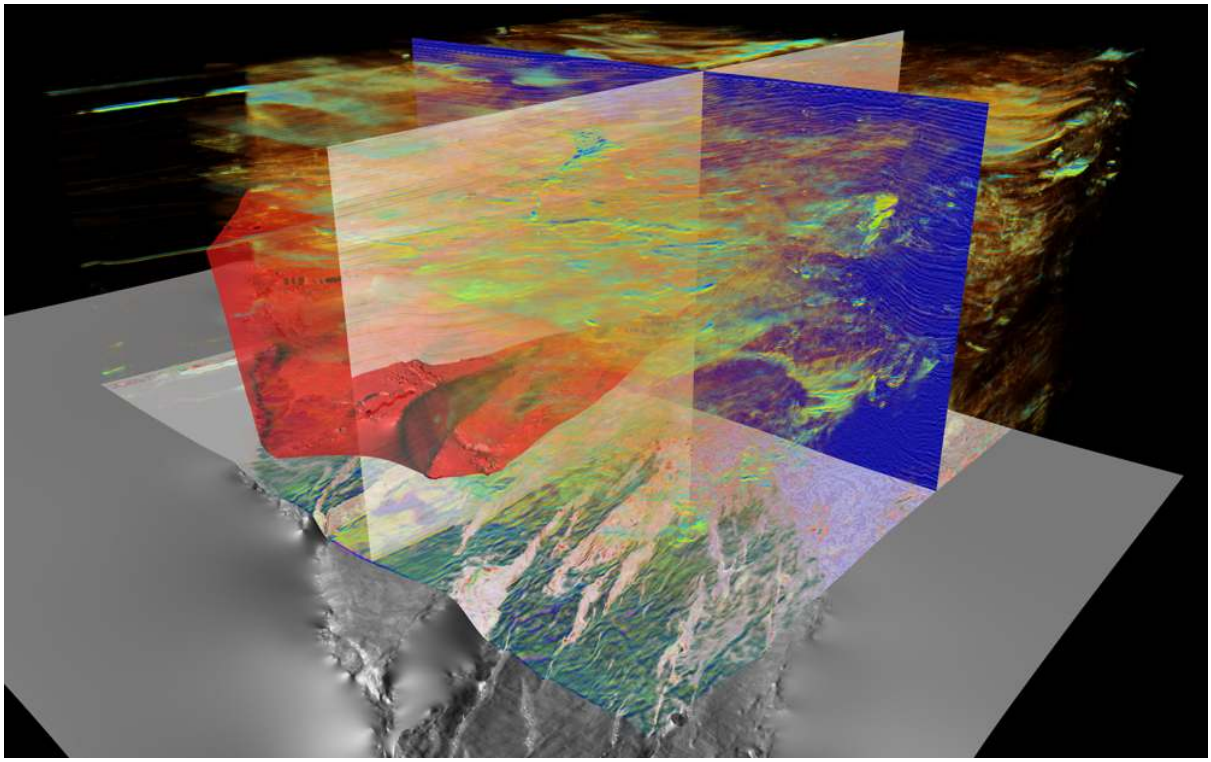

Figure 4.5

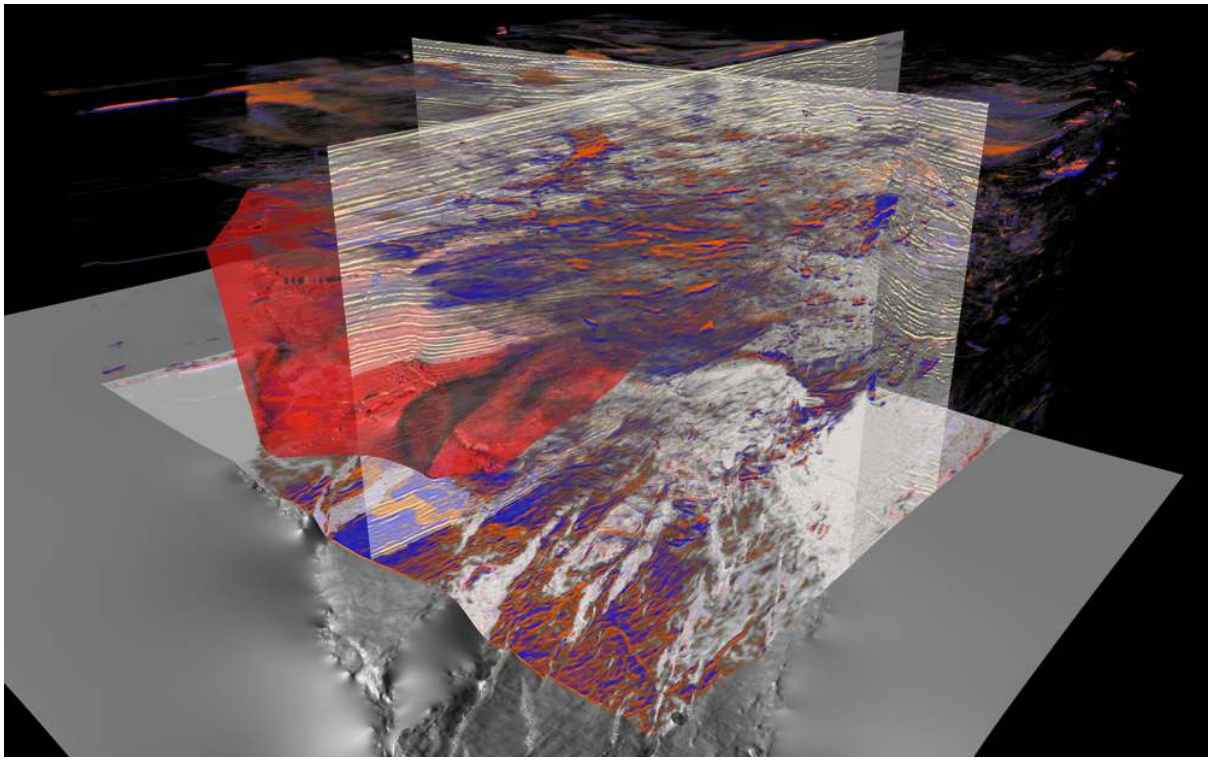

Figure 4.6

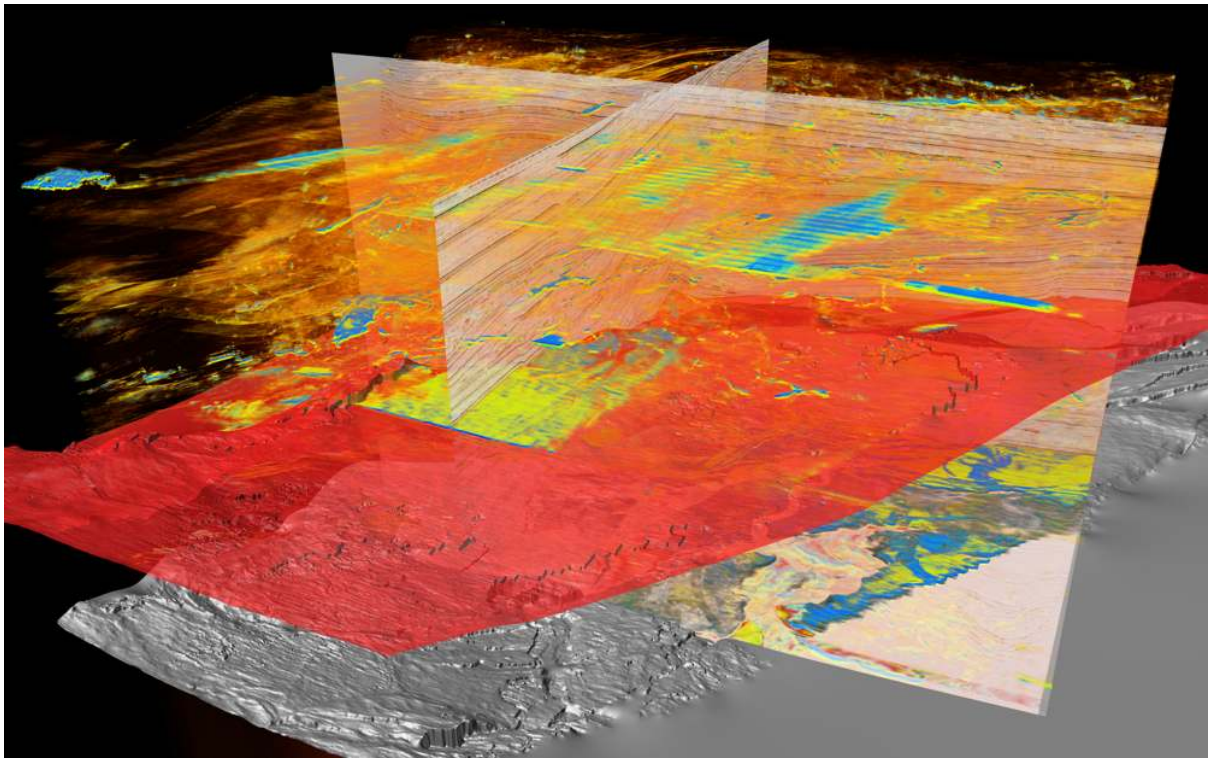

Figure 4.7

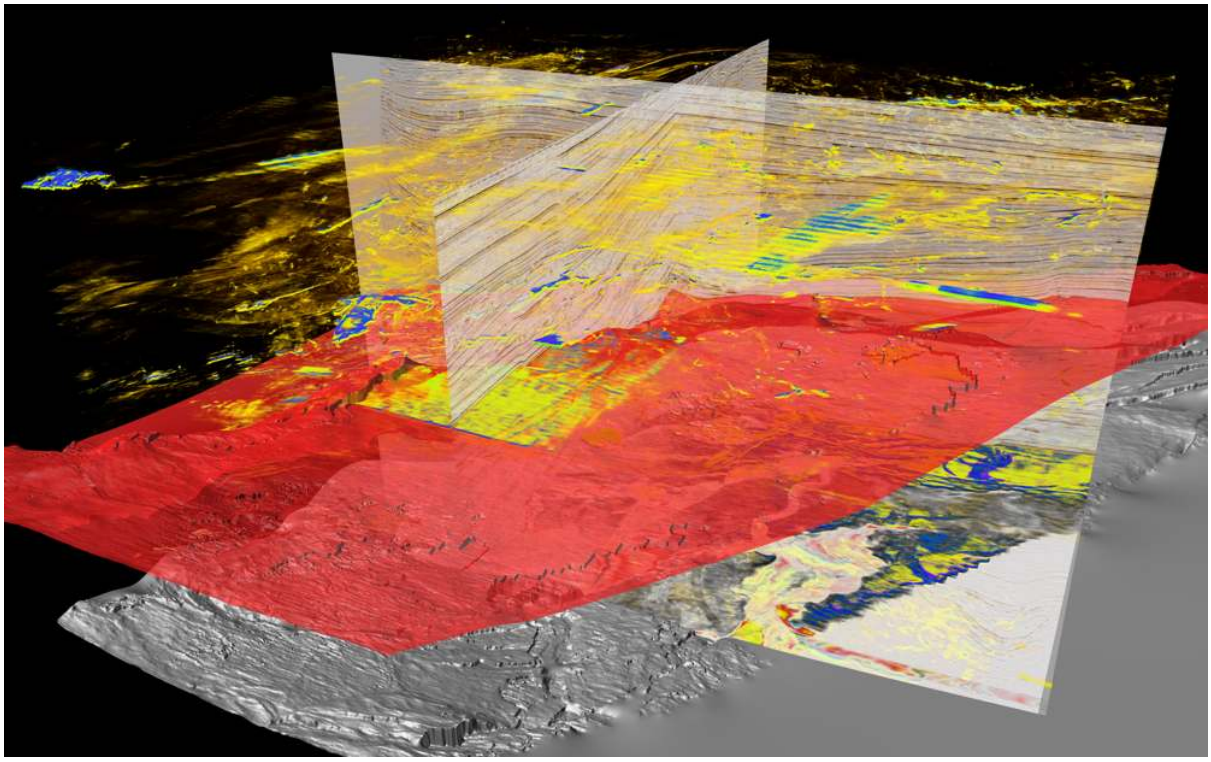

Figure 4.8

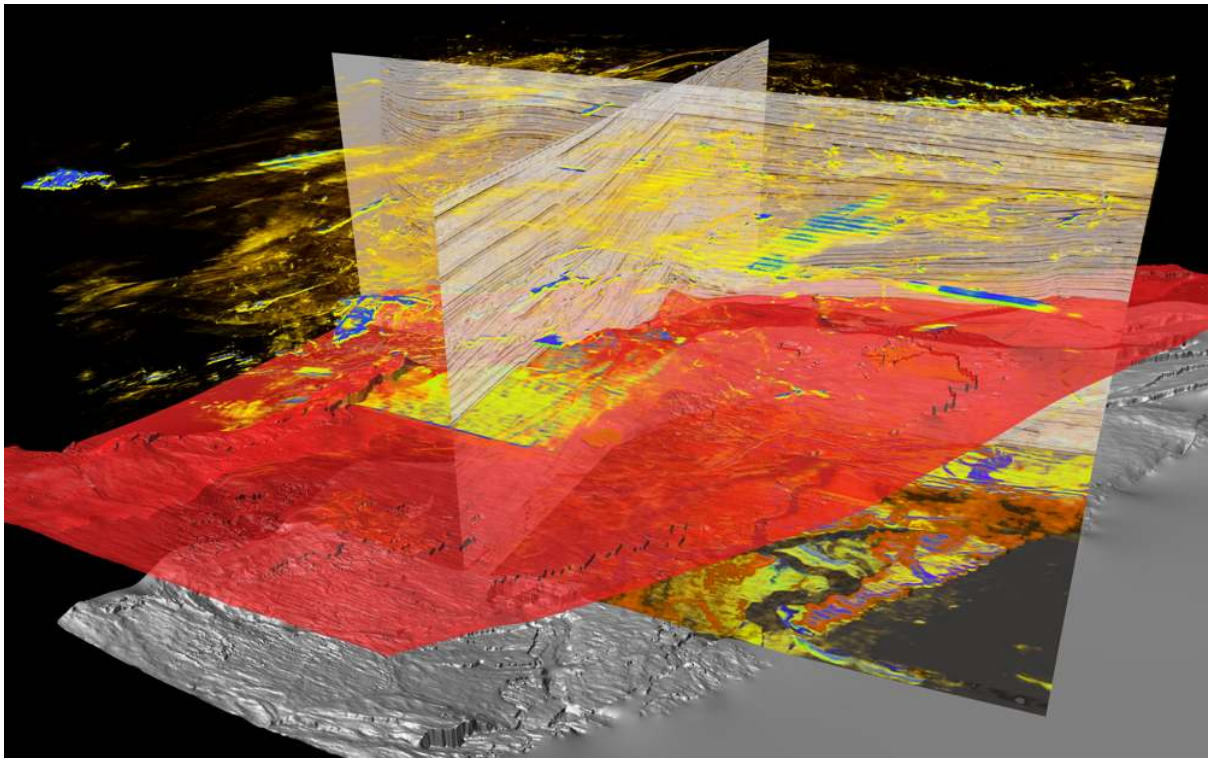

Figure 4.9

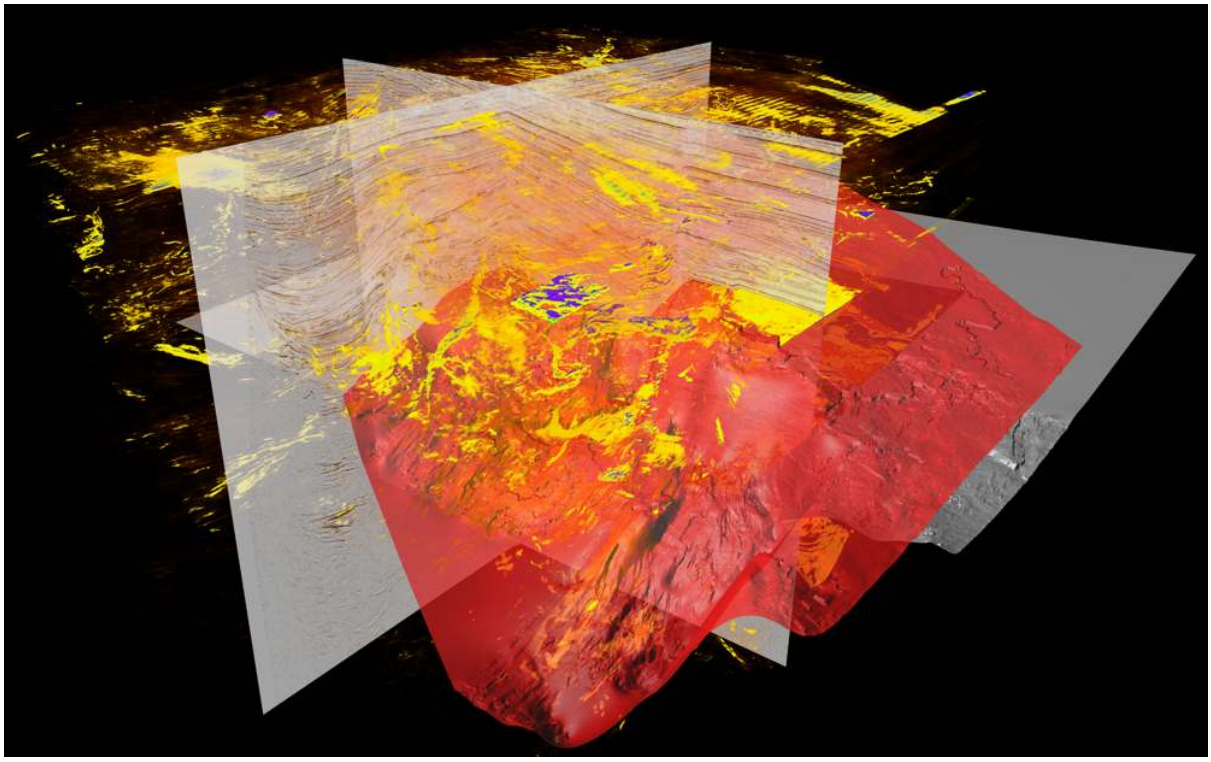

Figure 4.10

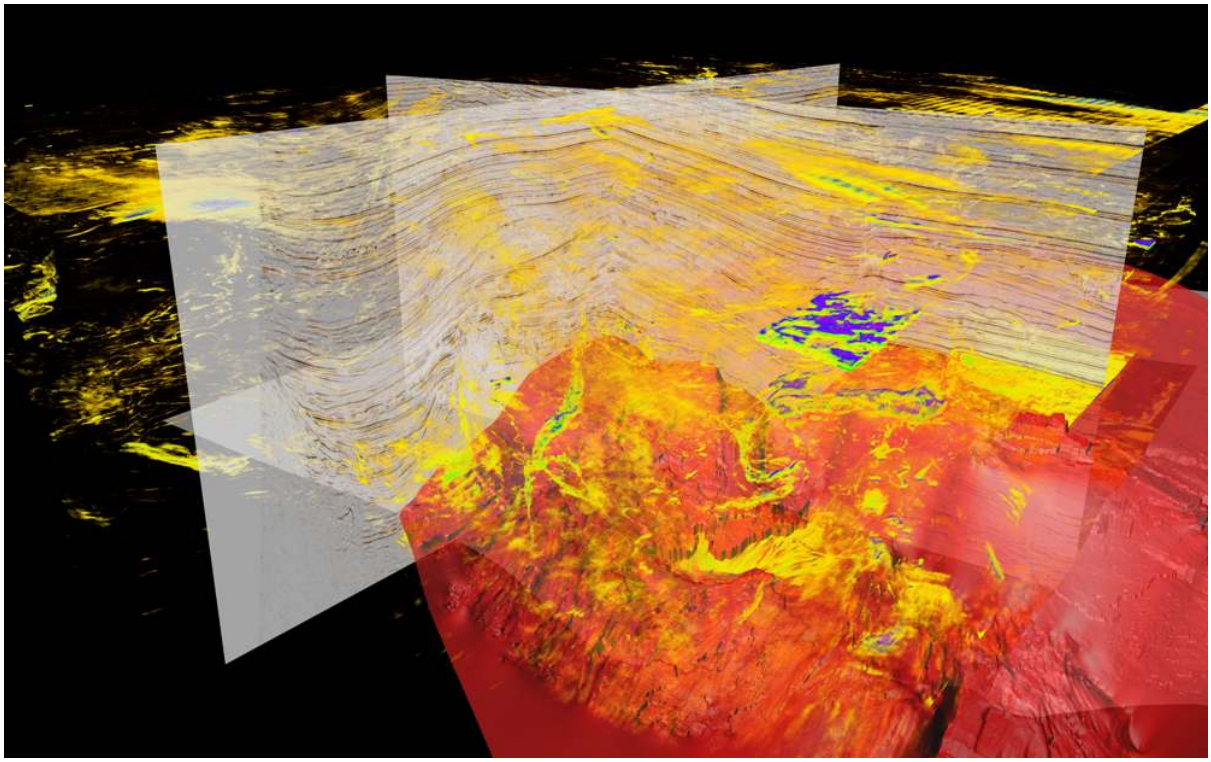

Figure 4.11

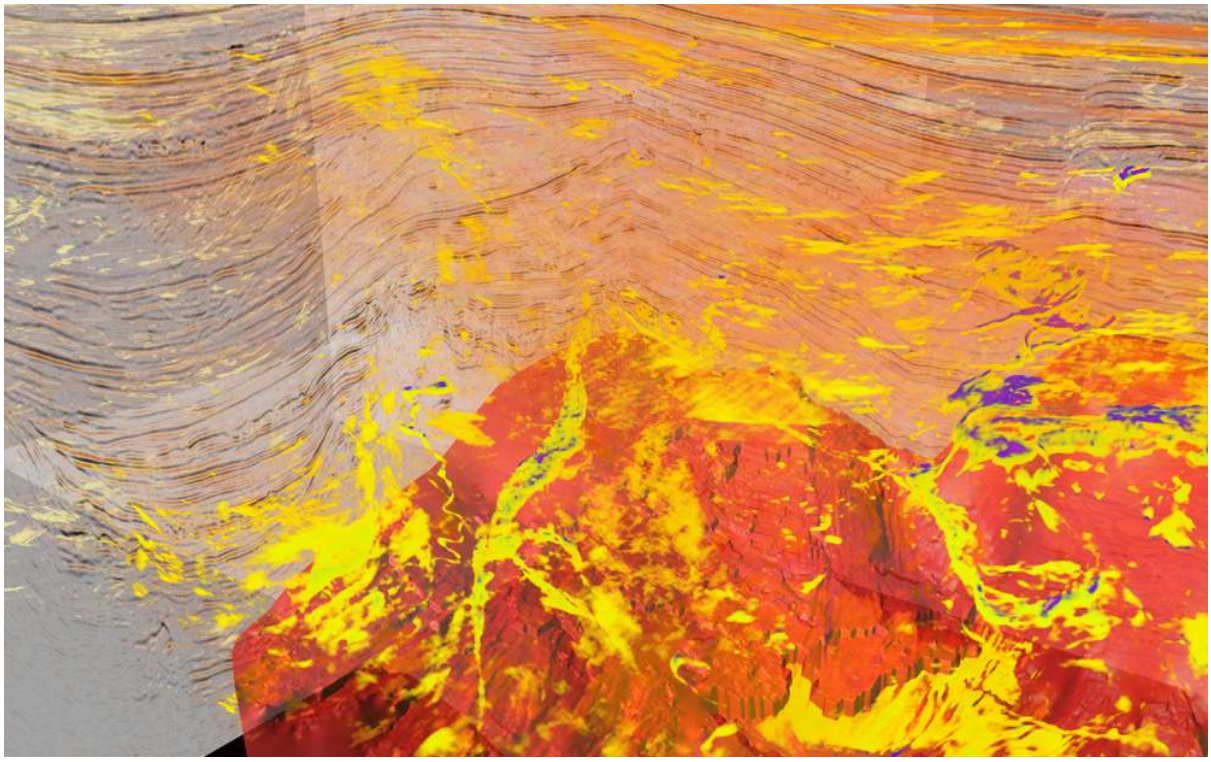

Figure 4.12

5 Glossary

Distributed Computing Environment (DiCE)
: A platform for writing parallelized high-performance computing applications. DiCE helps developers write applications that scale to thousands of multi-core CPUs combined with thousands of GPUs.

GPU cluster
: A group of GPUs on a set of hosts, treated as a unit for purposes of computation.

image tiles
: Non-overlapping rectangular regions of an image, each assigned to a *subcluster* to be rendered.

level-of-detail algorithm
: A representation method for large datasets using subsampled data in subregions currently deemed less important by the rendering algorithm (for example, those areas that are far away from the viewer in the three-dimensional space of the visualization). Because the dataset can be rendered at various degrees of resolution, this approach is also called a "multi-resolution model." Subregions may be dynamically replaced by the original data or by a subsampled representation as needed, though the resulting instability of the image can be distracting for interpreters viewing the visualization. Level-of-detail algorithms are used to address limited memory size when rendering large datasets, though at a cost of loss of detail. More fundamentally, the rendering of different levels might not represent the source data with complete accuracy, which in some application areas can reduce the interpreter's trust in the fidelity of the visualization.

multi-resolution model
: Another term for *level-of-detail algorithm*.

out-of-core algorithm
: Algorithms designed to process data too large to fit into the local memory attached to the processing units are generally called out-of-core or external memory algorithms. Such algorithms specifically manage the transfer of the required data between the different stages of the memory hierarchy to the actual processing units. Out-of-core rendering algorithms can be put into two categories: resource streaming algorithms and dynamic level-of-detail algorithms.

parallel rendering
: Most visualization systems can be classified based on the parallel rendering approach they follow, which is either sort-first or sort-last. In sort-first rendering, sets of screen-space pixels are distributed to computational units for processing; in sort-last rendering, the objects to be rendered are distributed. In general, the sort-first approach is ideal for high-resolution display systems but not suitable for large-scale data rendering. On the other hand, the sort-last approach is ideal for rendering large datasets but ill-suited for high-resolution displays.

proxy geometry
: Geometric or volumetric shapes that are able to display compute results, for example, using texture-mapping techniques. Proxy geometries include, for instance, three-dimensional planes (rectangular three-dimensional shapes), height-fields (height maps or terrains models), or voxel cubes (three-dimensional volumes).

resource streaming algorithm
: Resource streaming algorithms try to efficiently stream data required for rendering into GPU memory. They heavily depend on the efficiency of the transport of the data from one memory area to another. Slow transports result in large latencies between the start of the data transport and its availability for rendering, usually resulting in degraded rendering speed as the GPU is idling while waiting for the data.

scene element
: A geometric shape or a visual representation of data contained in the visualizations of NVIDIA IndeX, which can be modified with attributes and structured into *scene groups*.

scene group
: A hierarchical structure containing *scene elements*.

shape group
: An NVIDIA IndeX class from which an application can create a subclass to define a new shape by combining existing shapes and attributes.

staged rendering
: While processing the data of subregions available in GPU memory, the data of unavailable subregions is loaded asynchronously into GPU memory in place of already processed subregions. Each single GPU on a cluster node can hold a fixed number of subregions. These subregions are processed sequentially. After a subregion has been processed, it can be discarded from GPU memory, allowing the loading of a new subregion from main memory. Through overlapping this load process with the rendering of other subregions, NVIDIA IndeX can hide the memory transfer from main memory, allowing the rendering process to run uninterrupted, provided there are enough subregions in GPU memory to cover the load of the unavailable data.

subcluster
: A group of machines to which an image tile is assigned during rendering.

subregion
: A subset of a large-scale dataset produced by partitioning to permit processing of the subset on a single-cluster node.

volumetric data
: A correlation of data values to positions in three-dimensional space. Multi-valued volumetric data associates more than one type of data value with each three-dimensional position.