Data Issues

Lecture 11

Converting Data

- Two broad categories of data conversion
  - Lossless conversion
  - Lossy conversion

- Each of these categories occurs in two domains
  - File formats
  - Visualisation infrastructure

Lossless vs Lossy conversion

- Lossless conversion means that there is a bi-directional mapping between the formats
- Lossy conversion implies a non-reversible transformation - ie information is lost.
- Why use lossy conversions?
  - Lossy formats save space/bandwidth.
  - Visualisation often require format conversions and loss of information is of secondary importance.

Scientific File Formats

- There are many different file formats
- There is little standardization on scientific data formats
- The problem is everyone has different types of data.
- A few sophisticated scientific data formats exist that are extendible / self-describing
  - hdf (Hierarchical Data Format)
  - cdf, net-cdf (Common Data Format)

Scientific Data Compression

- Probably the majority of files that contain scientific data are uncompressed.
- Compression adds complexity to the file format and makes random access harder.
- Many scientific file formats are created as a “quick hack” to transfer data between systems. Inbuilt compression is just too hard.

Scientific Data Compression…

- Assuming compression is used. What type would it be?
- Lossless !!
- Common lossless compression algorithms include *Run Length Encoding* (RLE) which reduces repetitious symbol sequences to a symbol and a repeat count.

Image File Formats

- Images formats are marginally better organized.
- There are still around 50-100 “common” image formats most having several variations.
- “Image” formats fall into two broad categories
  - Bitmap
  - Vector.

Bitmap Images

- Bitmap images store each pixel of the image
- The result is a 2D array of values.
- Image file size is nominally constant regardless of the detail.
- Bitmaps can represent all types of images
- The resolution is limited by the number of pixels.

Bitmap File Formats

- Bitmap image formats include:
  - GIF
  - TIFF, GEOTIFF
  - BMP
  - JPEG

Vector Images/Data

- Vector data is defined by the basic elements making up an image/dataset:
  - points
  - lines
  - circles
  - polygons
  - etc
- A vector image is constructed using the predefined types.

Vector Images/Data

- Vector formats are best suited to simple images/data sets that have been constructed from simple types.
- They cannot be used to display arbitrary images.
- File size is dependent on complexity.
- Resolution is not a factor.
- Results are at displayed *device* resolution.

Vector File Formats

- Vector format include
  - Postscript (also supports bitmapped)
  - CGM
  - HPGL (generally plotting)
  - IGES
  - DXF (engineering 2D/3D data)

Image Compression

- Image compression is generally applied to bitmaps.
- Compression may be lossy or lossless.
- It is up to the user to determine an appropriate category.
- Most image compression algorithms are lossless.
- The most common lossy algorithm is JPEG.

Generic Data Compression

- A simple way to achieve data compression is to use a generic file compression tool found on most computer systems.
- All such tools are lossless.
- Examples include:
  - `compress` (unix),
  - `gzip, gz` (universal)
  - `zip` (mostly PC)
- Note: `tar` is not a compression tool

Visualization Conversions

Visualization Conversions

- Format conversions are always required at some point within a visualisation system.
- Some of these will be lossy conversions.
- Example:
  - Drawing to screen results in data being fitted into pixels. In general this is a lossy process. Pixels have limited colour range. Many raw data values may map into one physical pixel or one colour.

Visualisation Conversions…

- Other reasons for conversion include (*re) gridding*, discussed earlier.
- Data may not be supplied on a grid whereas many current algorithms require gridded data.
- Gridding data involves binning and averaging which are irreversible processes.

Downsizing Issues

- Downsizing is a lossy conversion, it is often needed on large datasets.
- Downsizing has its own set of problems
  - Cell size must increase if geo-referencing is important.
  - Aliasing (a sampling issue).

Surface Fitting Problems

- Algorithms which fit surfaces to data such as isosurface have several problems:
  - Limited resolution. The fitted surface is an approximation and may well be at a lower resolution that the original data which itself may be sampled…
  - Surface algorithms may not handle branches correctly. Bifurcating surfaces cause problems with algorithms like marching cubes.

Numerical Accuracy

- Calculations incur fractional *roundoff* error.
- Numerical resolution limits, eg:

```
1,000,000,000 + 1 => 1,000,000,000
```

- Similarly dividing numbers that differ by many orders of magnitude can cause problems. Sometimes simple rearrangement of the equation can avoid problems such as `NaN` and `INF`.

NULL Data

Missing Coordinate Data

- How do we identify that there is a missing coordinate.
- Obvious - leave it out !
- We can only omit from scattered (point) data or an unstructured grid.
- With a structured grid the coordinates are implicit and therefore always exist.
  - Eg can't "omit" a pixel in an image.

Invalid Coordinate Data

- But omission is not the same as a bad value
- Suppose a GPS system is required to log coordinates at several locations. If it loses a fix it may still have to give an answer.
- Need a method to specify an invalid / undefined value.
- ...

Invalid Node Data

- Similarly a coordinate may be defined but the node data may be invalid.
- Suppose we are obtaining values on a grid, measuring height of a surface every metre and go over a fissure.
- Bathymetric data often has bad values in it. Need to flag these during post-processing.

Invalid Data Solutions

- Can have a separate flag to identify a "bad value"
- The actual value is then ignored.
- This requires extra memory and can complicate programming.
- However it is probably the most robust solution.
- One benefit is that it preserves the "bad value" for later analysis.

Invalid Data Solutions…

- A common approach is to use an unlikely or otherwise invalid value.
- For example using -999 as a “bad value” for height above sea level.
- Using dates far in the future eg 9/9/99 !!
- This is a quick fix but can often lead to problems when the format becomes entrenched and used for extended purposes.
- The use/choice of NULL value is not clear.

Missing Node Data

- What is a suitable “missing value” value.
- First choice might be 0. A Pretty safe guess ?
- It depends...
  - Is 0 a valid data value. If it is problems *may* occur.
  - What if values are to be averaged for example ?
- The unknowns should be left out of the average calculations or biases occur.

Holes in Data

- One solution to a region of invalid data is to cut it out.
- Eg: an island may have no height defined along a deep fissure.
- A structured data set may be converted into an unstructured data set and problem cells simply omitted.
- This uses up far more memory and slows down drawing.

Displaying Undefined Data

- Undefined node values may be coloured a neutral colour such as grey.
- Systems which support per vertex transparency can have individual surface vertices made invisible, effectively creating a hole in a structured mesh.
- Problem can occur as the point still exists and is used in min/max calculations.

Undefined Data Processing

- Some software permits a null value to be defined.
- Any data values matching this null value are then omitted from calculations like min/ max and averages.