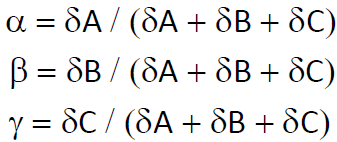
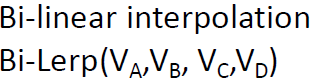
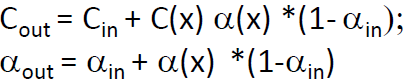
**Visual Analysis Pipeline** – **Data Acquisition**: medical imaging,**Data Preparation**: pre-processing,Feature **Extraction(rendering):Volume rendering** – Optical models,Transfer function.**Polygon rendering** – Raster graphics,Ray Tracing. **And Visual mapping(Analysis)**:**Topological**: Surface Topology- Contour Trees,reeb graphs. Vector field Topology (sink, source, saddle); **Geometry**: Surfaces and curves: first and second fundamental forms,various types of curvatures; **statistical**: -First,second,or higher order moments-Distribution,histograms-Entropy,information theory. (generic-isosurfaces,streamlines.Specific – vortices,material boundaries,Image) and **Animation**. Data sets are categorized into different types based on their underlying grid (domain structures). **Structured Grid**: –Consisting of a collection of points and cells arranged on a regular lattice–Every point in the structured grid can be indexed by (i,j) in 2D,(i,j,k) in 3D, etc.–The position of the points, and hence the geometry of the cells,can be either implicitly defined (Cartesian gird), or explicitly specified (rectilinear or curvilinear grid) •Cartesian mesh•Rectilinear mesh•Curvilinear mesh. **Unstructured Grid**: •Also called irregular grid data•Unstructured grid points are irregular located in space•It is often a result of space tessellation with simple shapes •Polygonal mesh•Tetrahedral mesh•Hybrid Mesh **Linear interpolation** is useful when looking for a value between given data points. It can be considered as “filling in the gaps” of a table of data. The strategy for linear interpolation is to use a straight line to connect the known data points on either side of the unknown point.  For triangle -   For rectangle - **Isocontours** – Points in a scalar field (eg. Pressure, temperature) that have a constant value. 2D = isoline. 3D = isosurface. Can compute isocontour in a cell based on linear interpolation.Find edges that are zero crossing, values at the two ends of these edges are greater and smaller than the contour value on either side. Calculate the position of P on each of the edges. (Compute intersection) Connect the points with a line. An isocontour intersect a linear rectangular cell? **16 times** **Four unique topological cases**: No intersection, intersect with 2 adjacent edges, intersect with 2 opposite edges, two contours pass through cell **Ray-Casting Algorithm:** For each pixel= cast a ray into the voxel and conduct linear interpolation to find data values that you then convert into optical properties (color and opacity), combine the optical properties before finding the final color.Shading: Compute a color for every sample in the volume.Classification: Compute opacity for every sample.Shading and classification are often computed through a transfer function.**Ray sampling**: Using discreet points along the ray that is travelling through the voxel, perform tri linear interpolation with these points to find the data values. Look up the transfer function to convert these to colour and opacity. Composite the colour and opacity B2F compositing  F2B compositing Flow Line Visualization - **Streamline**: a field line tangent to the velocity field at an instant in time.Release a particle into the flow and perform numerical integration to compute the path of the

particle.**Pathline** : the trajectory of a massless particle released from a seed point over a period of time. This is similar to what you see when you take a long exposure photograph of car lights on a freeway at night**Streakline** a line joining the positions, at an instant in time, of particles released from a seed point Continuously inject particles into the flow at each time step and track the paths of the particles**Timeline** a line connecting a row of particle that are released simultaneously. Timelines are generated by injecting rows of particles at some fixed time interval In a steady flow field, streamlines, pathline , and streaklines are identical. However, they can be very different in unsteady flows.Euler’s method  **Particle Tracing algorithm**:**1.** specify seed position p(0), t =0 **2.** Perform cell search to locate the cell that contains the p(t).**3.**interpolate the velocity field to determine the velocity at p(t) **4.** Advance the particle from p(t) to p(t+δt) using a numerical integration method **5.** Repeat step 2 until the particle moves a certain distance or goes out of bound. **Notes on Particle Tracing**•The accuracy of particle tracing depends highly on the step size and the integration method•Flow solvers are often second order accurate in time, so particle tracing method should be at least third order or higher•The velocity data need to be interpolated between two consecutive time steps, which can introduce errors too

**Visualization**: Help people carry out tasks more effectively. Needed to augment human capabilities, not replace people with computational decision-making methods. **Resource limitations**: Computational limits (processing time and system memory), human limits (attention, memory), display limits (limited number of pixels, information density ratio to whitespace). What, why, how? Nested model: **four levels of visual design**.**Domain:** Who is the target user? **Abstraction**: What is shown and why is the user looking at it?**Idiom**: How is it shown?**Algorithm**:Efficient computation.**Why is validation difficult**?**Domain**: Misunderstood users needs. **Abstraction**: Showing the user the wrong thing. **Idiom:** The way you are showing the visualisation does not work. **Algorithm:** Code is too slow.**Soln to validation being difficult: Domain:** Observe target users using existing tools.**Idiom:** Justify design with respect to alternatives. **Algorithm:** Analyse computational complexity. **Idiom**: Analyse results qualitatively. **Abstraction:** Observe target users after deployment. **Domain:** Measure adoption.**Three** **dataset types**: tables(items, attributes), networks(Links, items, attributes), spatial (fields: grids, positions, attributes, geometry: items, positions). **Attribute Types**: categorical(F.name,L.name),Ordered (Ordinal,Quantitative). **Ordering Direction**- Sequential,Diverging,Cyclic

**Three categories of actions**:**Analyse**: Consume: discover, present, enjoy. **Produce**: annotate, record (save each step of interaction on the visualisation tool), **derive** (derive new data from tool to show). **Search**: Do users know the target or the location? **Query:** Identify, compare, summarise. **Targets:** To know what portion of the dataset you need to use for the visual design. All data required if analysing trends, outliers, features. Network data to understand topology and paths. Spatial data for shape comparison. **Marks:** ○ Geometric primitives○ Different spatial dimension◉ Channels (visual variable): ○ Control appearance of marks ○ Can redundantly code with multiple channels. **Visual Encoding**- Data/values – shapes. Analyze idiom structure: as combination of marks and channels. **Mark** represents data, **Channel** represents attribute. **Colour**: Decompose into 3 color channels**: Luminance & Saturation**(ordered), **hue**(categorical). **RGB**: Good for display hardware. Bad for visual encoding.**CIE L\*a\*b\*:** good for visual design. L: lightness frm 0-100. a\*green-red b\*blue-yellow. **HSL**: Better but L = lightness not luminance. Luminance: Used for detecting edges. Legible text & fine-grained detail requires luminance contrast**. Virdis color map**: colorful, perceptually, uniform, colorblind- safe, monotonically increasing luminance. Rainbow: Bad default for continuous/ordered data. Unordered and nonlinear. May perceive edges due to rainbow instead of data. **Pro**: Fine-grained structure visible and namable. **To design for color deficiency**: Avoid encoding by hue only, vary luminance, change shape. **Color channel interaction**: Size affect salience, small regions need higher saturation. **Categorical**: Aim for maximum distinguishability. **Sequential:** Ramp luminance or saturation. **Diverging**: Useful when data has meaningful midpoint. **Neutral color** - midpoint. **Saturated colors**- end points. **Cyclic**: date, time, month. **Selecting a color map:** Segmented or continuous? Diverging, sequential, cyclic? Single or multiple hue? Colorblind safe? **Scatter Plot**: No keys. Channels = horizontal and vertical position. Tasks = Find trends, outliers, distribution, correlation, clusters. Scalability = 100s of items. Pros: Depict large amount of data, clarify trends better than tables, easily compare data, not effected by extreme items. Cons: Subjective interpretation. Not suitable for representation of more than 2 variables. Only rough idea of correlation. **Bar Chart**:1k ,1val. Channels: length (express value). Spatial region: 1 per mark, separated horizontally, aligned vertically, ordered by quantitative attribute. Task: Compare values. Scalability: Dozens to 100s of levels for key attribute. **Line/dot chart**: 1k, 1val (quantitative attributes). Channels: Aligned lengths for quantitative value. Ordered by key attribute into horizontal regions. Task: Find trends. Scalability: 100s of key or value levels. Don't use for categorical key attributes bc violates expressiveness principle. Bar chart if categorical. Line chart if ordered. **Dual axis line chart**: Users focus on: crossing point (does it convey proper info?), correlation/trend/rate. Not intuitive to know which line belongs to which axis so use different colours for pairing. Confusing. **Stacked bar chart**: 2 categorical/ordinal attr, 1 quantitative attr. Mark: vertical stack of line marks, Glyph: composite object, internal structure from multiple marks. Channels: Length and color hue. Spatial regions: one per glyph. Task: Part-to-whole relationship. Compare bars at bottom (close to the baseline). Several to one dozen levels for stacked attribute. Normalised : Stacked bar chart, normalized to full vertical height , task: compare ratio among same categories.**Streamgraph:** 1 categorical k, 1 ordered k, 1 quantitative val. Task: Part-to-whole relationship over x-axis. Hundreds of keys (x-axis direction), Dozens to hundreds of keys (y-axis direction).**Gantt:** 1k, 2val. 1 categorical, 2 quantitative. Task: Emphasize temporal overlaps, start/end dependencies between items. Dozens of key levels. Hundreds of value levels.**Slopegraph:** Channels: 2 vertical position: express attribute value, linewidth/size, color. Task: Use the “slope” to emphasize changes in rank/value. Hundreds of value levels. **Heat Map**: 2 categorical, 1 quantitative. Marks: area. Separate and align in 2D matrix. Channel: color by quantitative val. Tasks: find patterns, clusters, outliers. 1M items, 100s of categorical levels, ~10 quantitative attribute levels **Scatterplot Matrix:** Dozen attr, dozens-100s items. Pros: easy to understand.Cons: scalability -The number of the sub-scatter plots grows exponentially. Used for multivariant data. **Parallel coordinates**: Positive correlation=Parallel lines, Negative correlation=lines cross at halfway point, Uncorrelated= Scattered crossings. Order of axes matter. Pros: scalability - It can show much more variables. Cons: Most people know how to interpret parallel coordinate plot. People need to be trained to interpret it. Use **radial layout** for audience to get general sense of part to whole relationship of data, precision is less important.**Pie chart:** visually appealing, ok for part to whole judgements. Alternative visualisations better than radial layout to understand data. Angle/ area less accurate than line length. humans are better at comparing lengths than angles.**Radar plot**: Not so proper when categories are not ordered/cyclic. **Arrange Spatial Data (Map)** Geographic visualisation: Maps advantage = people are familiar with locations on a map, Visually encode given spatial geometry as marks using 2D position channel. Marks: area, shape, position. **Choropleth map:** Understand spatial relationships. Geographic geometry. 1 quantitative attr/ region. Encoding region and color. Cons: hard to show multiple attr, large area attracts more attention, often not suitable for showing total values. **Symbol Map**: Good alternative to choropleth, encode multiple attributes of a region, size channel represents attribute, keep original spatial map in background, symbol represents data. Intuitive to read, marks: symbol size based on attr, glyphs: symbol size can be uniform. Cons: overlap, complex glyph use requires explanation.**Cartograms**: Distort shape of geographic region so area directly encodes data variable. Goals:Maintain constraints of relative position and contiguous boundaries with neighbours, try resemble original shape. Can distort visual channels such as shape/ position which used to recognise regions, hence important to find proper resolution.**Dot Density:** Visualize “distribution” over the map. 1 symbol represents 1 obj / constant. See spatial patterns/ clusters. Straightforward, avoid choropleth non-uniform region size problem. Difficult to extract quantities, Performance disadvantage: rendering a lot of points could be slow. **Arrange Networks and Trees:** Network: Model relationships between things. **Tree**: Special case of network with no cycles. Network tasks can be topology or attribute (find distributions) based. **Good Node Link Diagram**: Minimise: edge crossing, distance between topological neighbour node, total drawing area, edge bends and length disparities. Maximise: angular distance between edges. Emphasise symmetery.

**Force-directed layout**: Model: Link-Springs pull together. Nodes-Magnets repulse apart. Minimizes overlaps in the graph, evenly distributes nodes and links, and organizes items so that links are of a similar length. It is clear, reliable and makes a good all-rounder for any type or size of dataset, because the focus is on finding patterns and symmetries. Tasks: Explore topology, locate path, cluster. **Restricted layout: Circular/ Arc:** Layout nodes around circle/along line. Node ordering to avoid edge crossings clutter. **Adjacency Matrix View**: Data=Network. Transform into same encoding as heatmap. 1 quantitative attribute (weighted edge between nodes). 2 categorical attributes(Node list\*2). Cell shows presence/absence of edge. Scalability: 1k nodes 1m edges. **Node-Link diagram**: Topology understanding, path tracing, intuitive flexible, no training needed. Best for small networks. **Adjacency Matrix**: Focus on edge inst of nodes, layout straightforward, predictability, scalability, some topology task trainable. Best for large networks tht don't involve path tracing. **Node-link tree**s: Clear parent/child structure, Compact without overlap, Rectilinear and radial variants. Distance from root: depth in the tree. Angular (radial), horizontal(regular) proximity: siblings. Tasks: Understanding topology, following paths. Regular: several dozens ~ hundreds nodes, Radial: 1K – 10K nodes. **Two ways to represent links**: Connection: All node-link diagrams, emphasise topology, path tracing. Networks and trees. Containment: All treemap/sunburst/icicle variants, emphasise attrs at leaves, only trees. **Treemap:** Quantitative attr at leaf nodes. Area containment marks for hierarchical structure, rectilinear orientation, size encodes quantitative attr. Tasks: query attr at leaf nodes. Up to 1M leaf nodes.**Consider when creating a tree**: Avoid wasting space, where to fit labels, rectilinear v radial layout, tree depth, link relationships, connection or containment for links. **Handle Complexity: Manipulate** Handle Complexity: If not working, use 1 static view 2 solve prblms if not 2 complicated 2 understand. If data 2 complicated dnt use 1 static view 2 solve all problems. Change view over time, facet across multiple view, reduce item/attribute within single view. We can change: parameters, alignment, visual encoding, rearrange order. **Change:** Re-encode: Diff idioms serve diff tasks. Users wnt to complete diff tasks frm ur tool. ∴ re-encode (diff idioms) same data by users need (interaction). Eg. Choropleth map -> bar chart (observe population distribution across USA -> observe state w highest population). **Change parameters**: Add widgets 4 users to control which subsets of data r visualised. Eg. sliders, buttons, radio buttons, checkboxes, dropdowns. By controlling users know whats happening, self documenting. Con: Takes up space. **Reorder:** Find extreme values/trends. Observe attribute correlaton better.**Realign**: Aligning 2 diff segment -> flexible comparison. Eg. In stacked bars easier to compare first segment or total bar.**Animated transition**: smooth interpolation between states. Supports item tracking. Understand whts happening between 2 states w/o explanation. Eg. Hierarchical bar chart. **Manipulate**: **Select**: Basic operation for most interaction. eg. **Tooltip:** Select point on line chart to reveal text providing more detail. **Design choices**: selection types? Interaction types, eg. Click v hover, click types (shft click, optn click). Application semantics? Eg. Adding to or replacing selection, add/delete items. **Highlighting selection**: Change selection colour/size/shape, add outline. **Navigate:** Change viewpoint, eg. Rotate/zoom/pan(translate). Eg. Google maps. Semantic zoom: When zooming away, instead of seeing a scaled down version of an object, see a different representation, eg. G maps. Utilize pixels to put more info/change visual encoding. Geometric: maintains similar physical properties. **Scrollytellin**g: Navigate page by scrolling. Pros: familiar, intuitive, linear. Cons: Scrolljacking, unexpected behaviour. **Unconstrained**: Users freely move virtual camera. Easy for designer to implement, hard for user to control. **Constrained:** Often uses animated transitions, trajectory computed depending on selection.

**Interaction**: Pros: adv of comp based v paper based visualisation, flexible/powerful/intuitive, fluid task switching (diff visual encoding supports diff tasks), animation gives support. Cons: Users don't interact as planned, invisible functionality diff to discover, time cost.

**Path tracing** is a ray tracing algorithm that sends rays from the Camera and, when a ray hits a reflective or refractive surface, recurses the process until it reaches a light source. The series of rays from the Camera to the Light forms a "path".