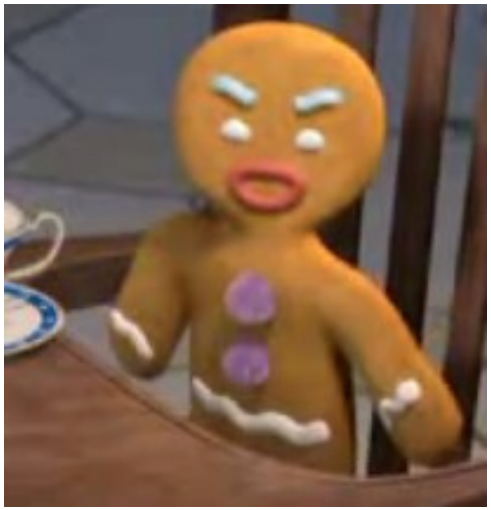
Course Review

Computer Animation and Visualisation

Taku Komura

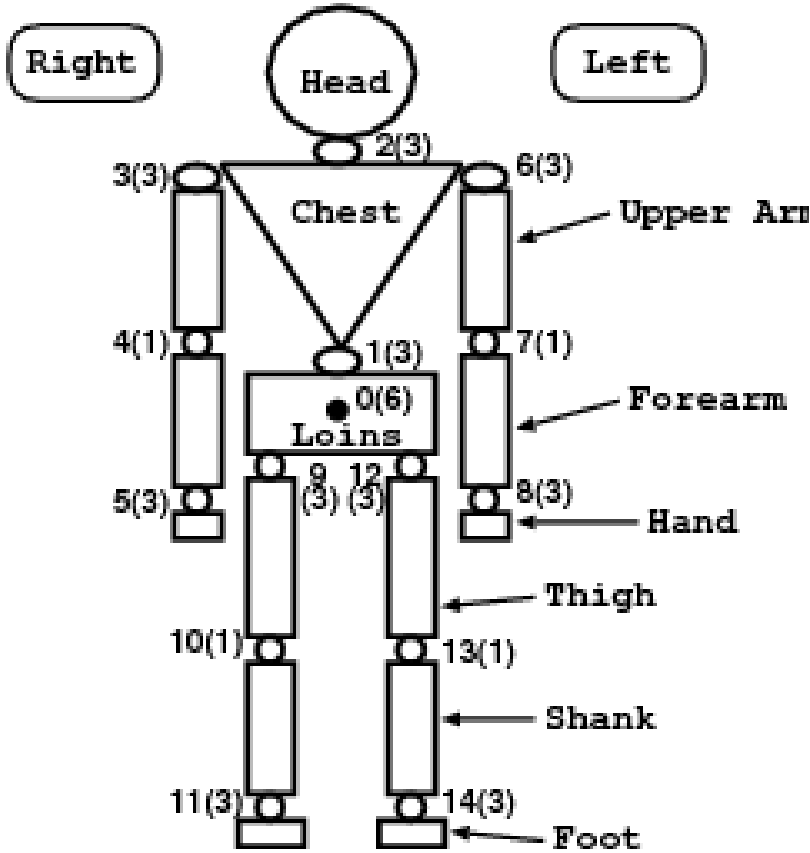
Characters include

- Human models
- Virtual characters
- Animal models



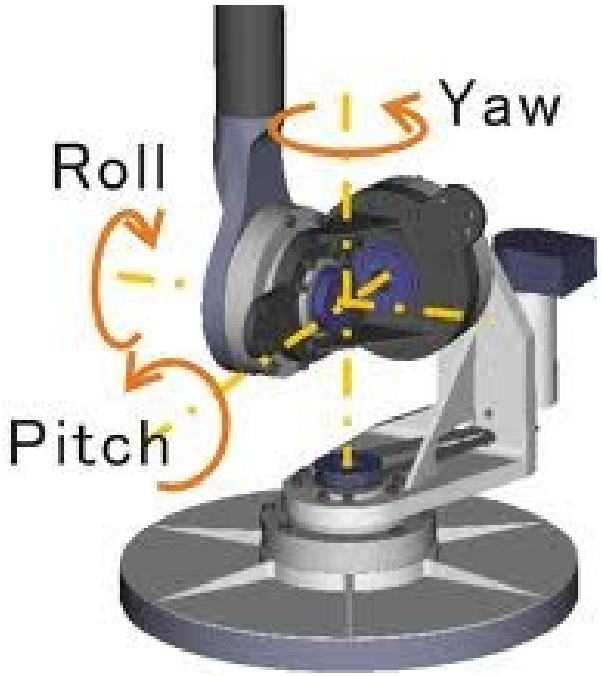
Representation of postures

- The body has a hierarchical structure
- Many types of joints



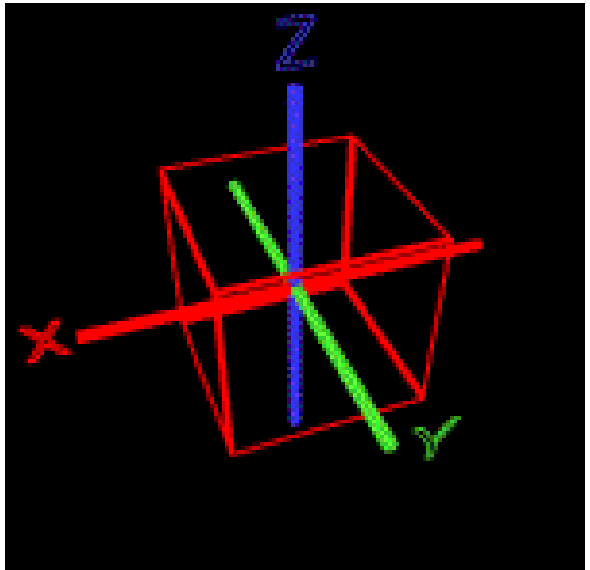
Gimbal joint : Euler Angles

- 3DOF joints
- Comes from Robotics
- 3DOF joints in robots were designed by connecting three motors pointing different axes



Problem: Gimbal Lock

- Two rotational axis of an object pointing in the same direction - 1DOF is lost
  - For example for rotation defined in the order of X-Y-Z
  - Gimbal lock occurs when rotating Y for 90 degrees.
  - X and Z axis get pointed down the same axis



Capturing human motion

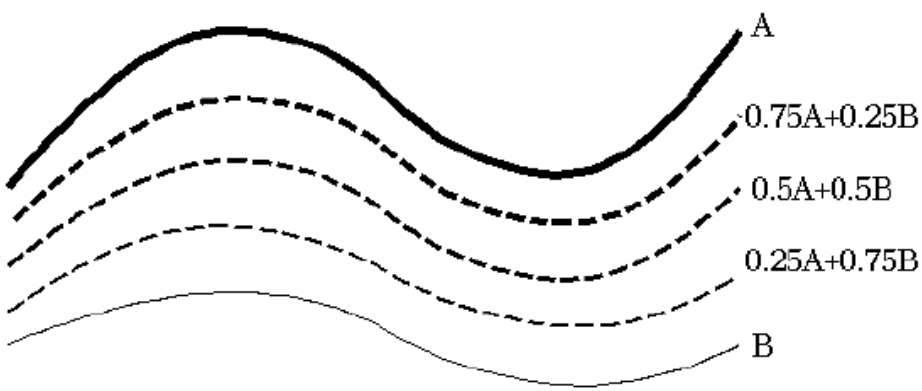
- We use the motion capture device (Mocap)
- There are four major types of Mocaps
  - Optical
  - Magnetic
  - Inertial trackers
  - Mechanical

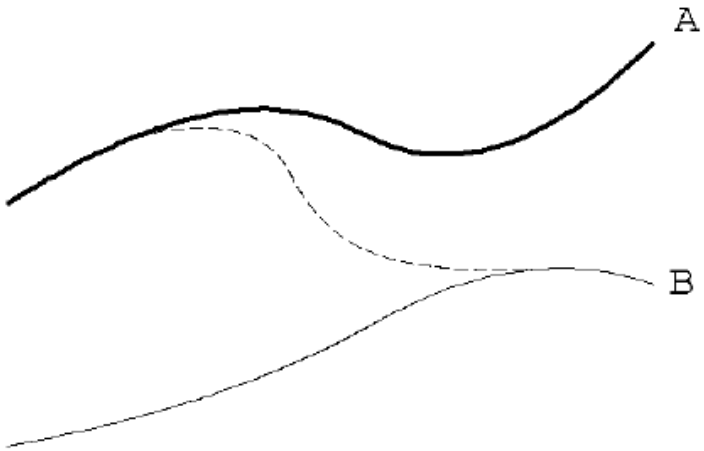
Motion Blending

- Given two different motions A and B

  Blended Motion = A * (1-s) + B *s (0<s<1)

  - How can we use this?
    - Generate a motion in between
    - Gradually shift from motion A to motion B
    - Concatenating two motions – blending the two ends by gradually shifting s from 0 to 1

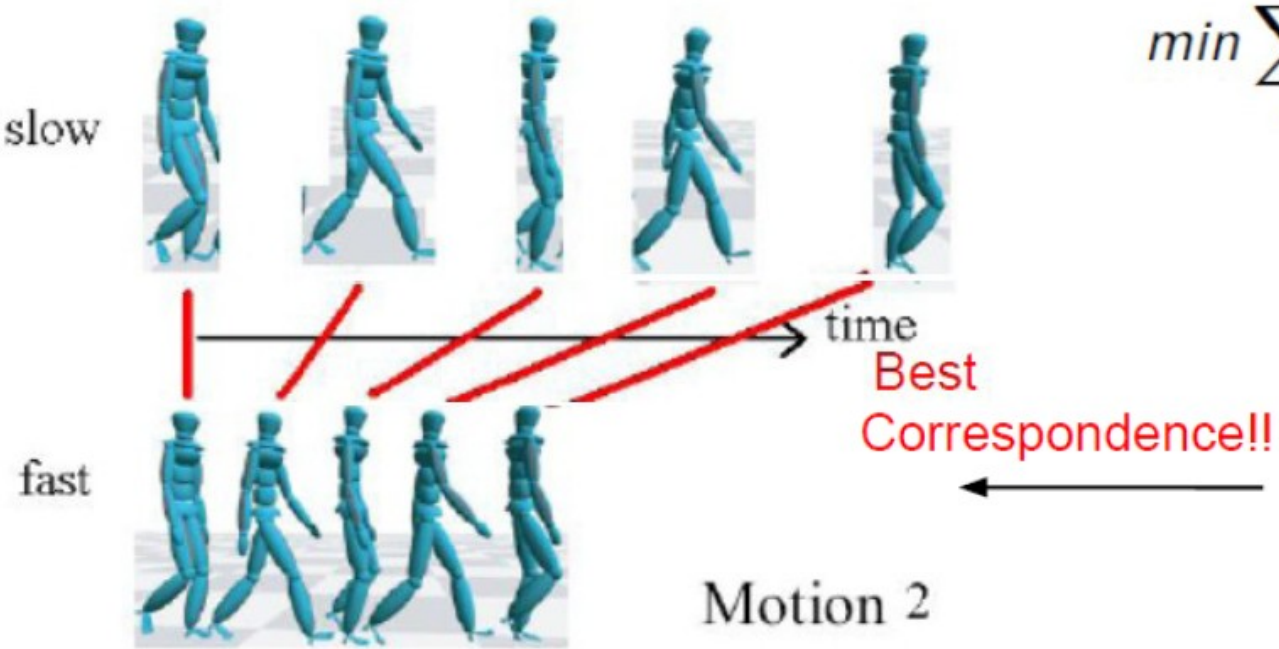


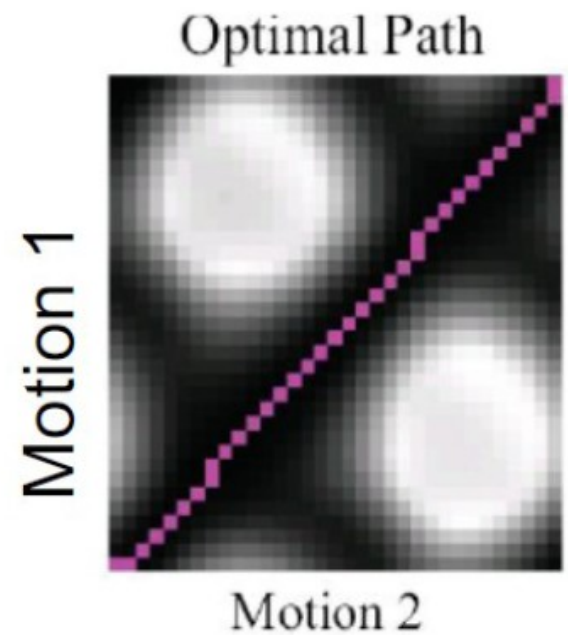


Dynamic Time Warping (DTW)

- Define difference of postures at every frame in motion 1 and motion 2
- Create a similarity matrix of all frames
- Find the shortest path from the left bottom to the right top in this matrix (assume the matrix is a map) - this gives

$$min \sum_i d_i(q_1, q_2)$$

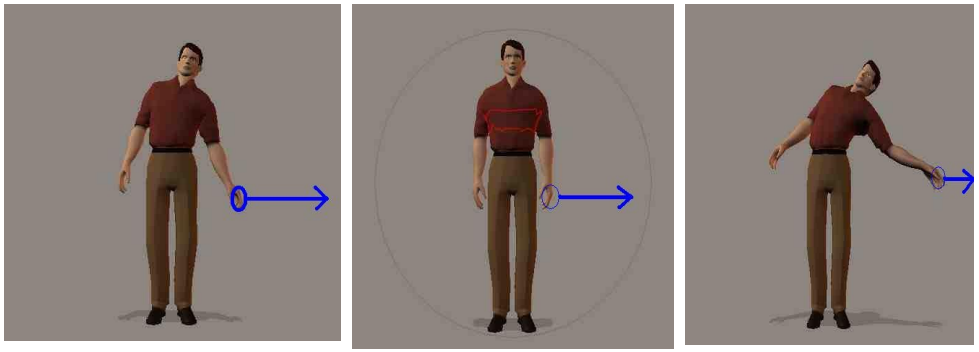


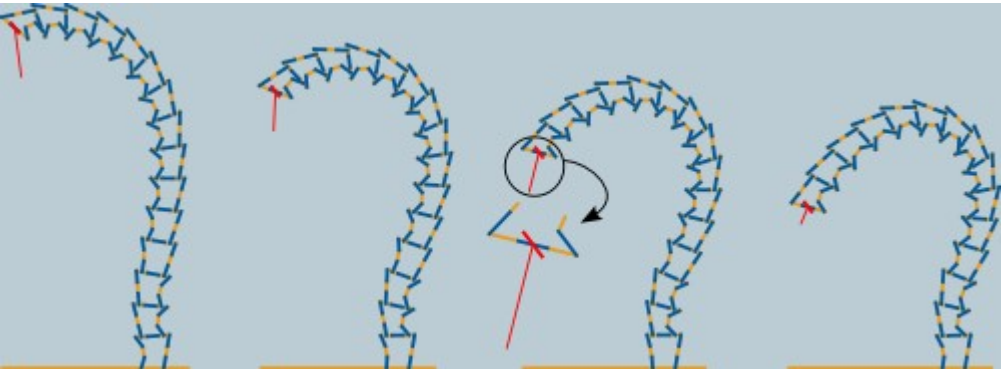


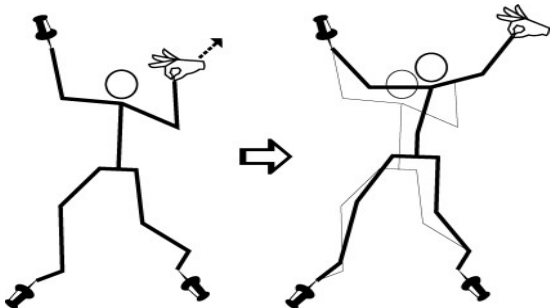
http://www.youtube.com/watch?v=EL7ARaH5jHU

Inverse Kinematics

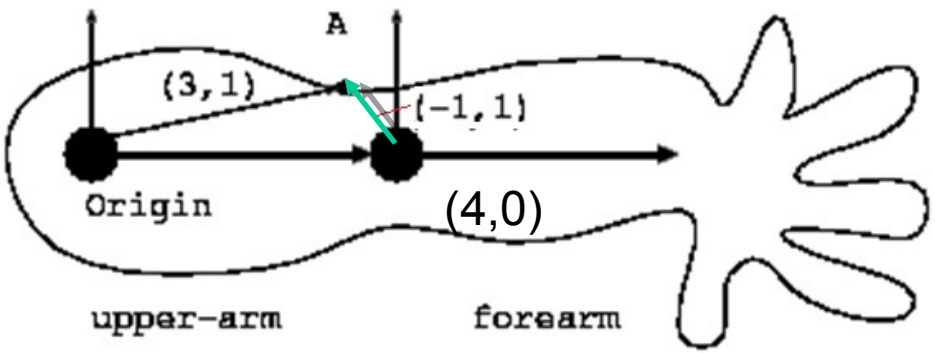
- Editing the postures/motions by specifying the 3D location of the avatars
- We can use IK to solve the problem of foot sliding
- Edit the postures, frame by frame, and drag the foot to the original position



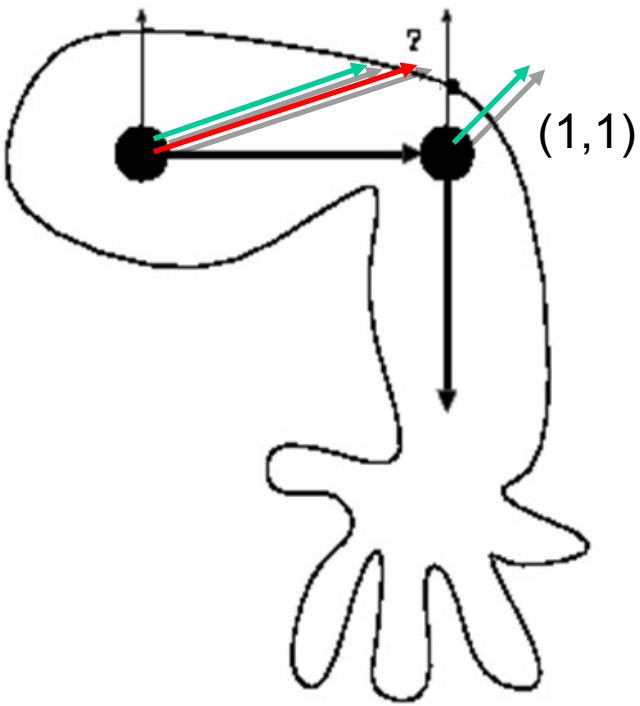




Linear Blending



Computing the vertex location based on each bone, and then blending the results

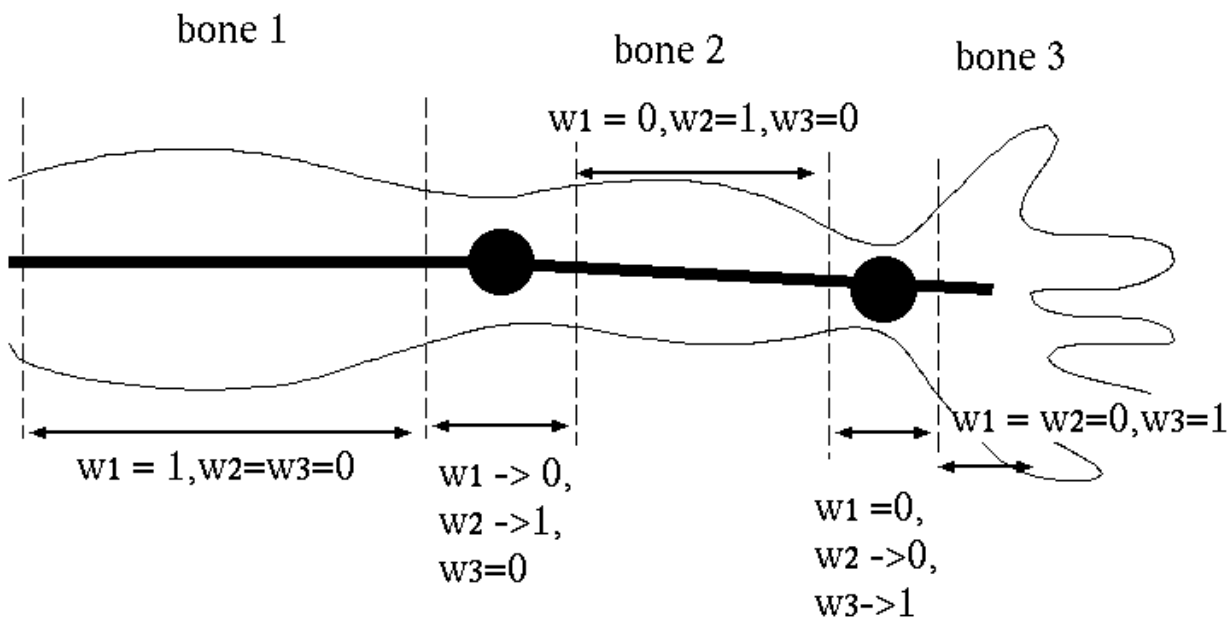


$$v_i = \sum_{i=1}^{b} w_i M'_i M_i^{-1} v_g \qquad \sum_{i=1}^{b} w_i = 1$$

How to decide the weights?

Decide the mapping of the vertex to the bone

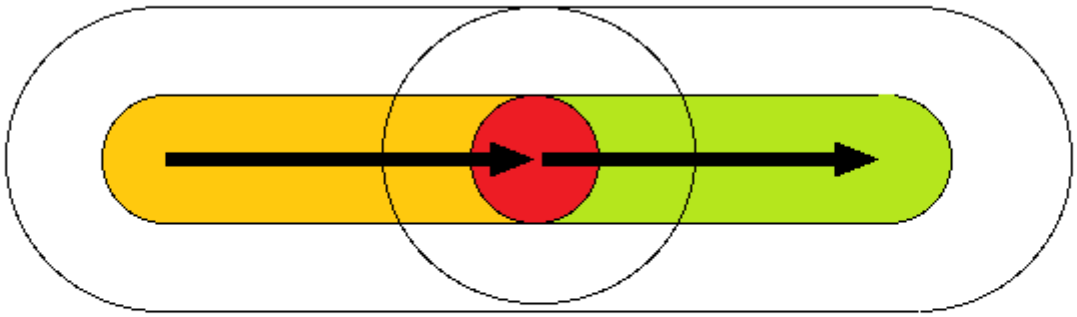
- If vertex v is in the middle of bone i, then $w_i = 1$ and for the rest $w_{j \neq i} = 0$
- If the vertex is near the border of bone i and i+1, *wi* will gradually decrease to 0 and *wi+1* will gradually increase to 1
- If the vertex is affected by more than three bones, the weight can be determined according to its distance to each bone



How to decide the weights?

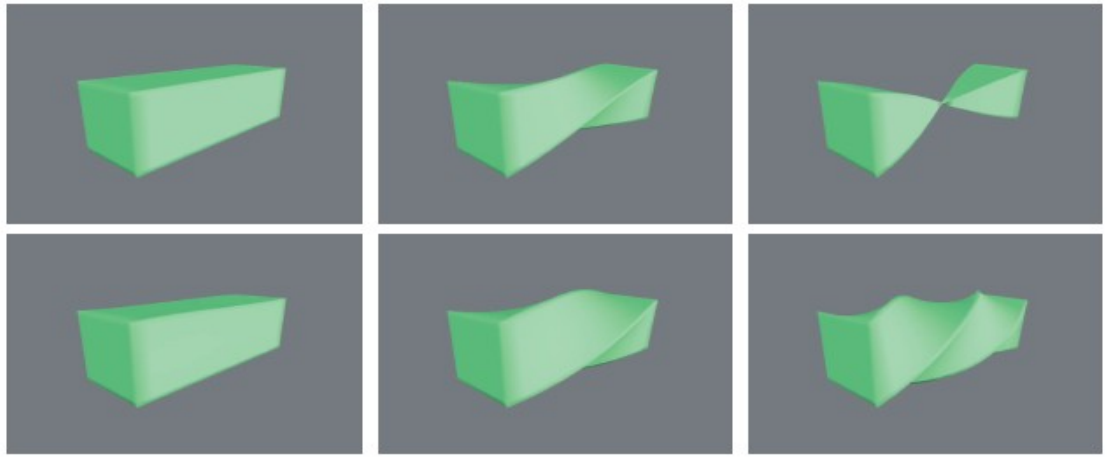
Example: use the Euclidean distance

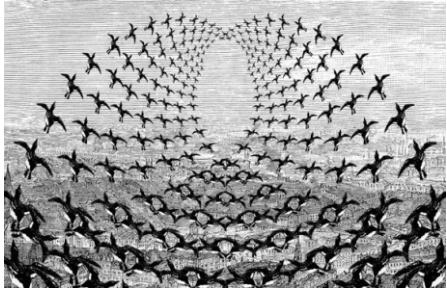
- Surround the bones by the inner and outer capsules
- If the vertex is inside only one inner capsule, the weight for the corresponding bone is 1, and the rest are 0
- If inside multiple inner capsules, compute the distance to each bone, and use that to decide the weights (longer distance, lower weight)



Problems with Linear Blending

- The meshes exhibit volume loss as joints are rotated to extreme angles.
- These are called “joint collapse” and “candy wrapper” effect
- Simple linear blending in the Cartesian space causes the artefacts

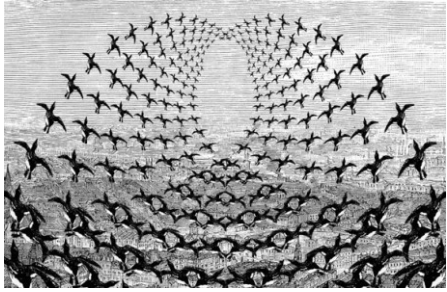




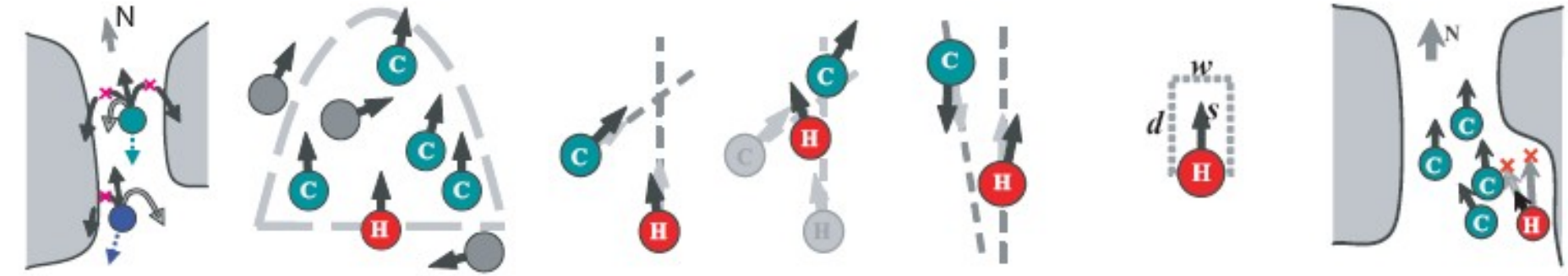
Flocking Models – Most basic agent model (Reynolds ’87)

- The agents interact based on simple dynamics
- Good to simulate
  - Flock of birds flying,
  - School of fishes swimming

Collision avoidance

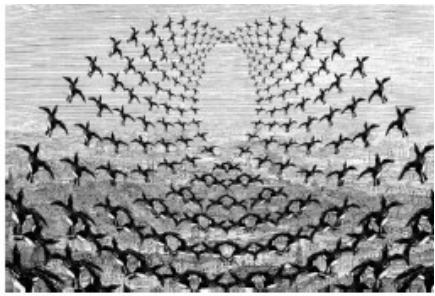


- Humans avoid others in streets in a complex way
- Sometimes wait, sometimes, move aside while walking
- Need to either
  - Model them based on a complex collision avoidance engine
  - Find a good mathematical model

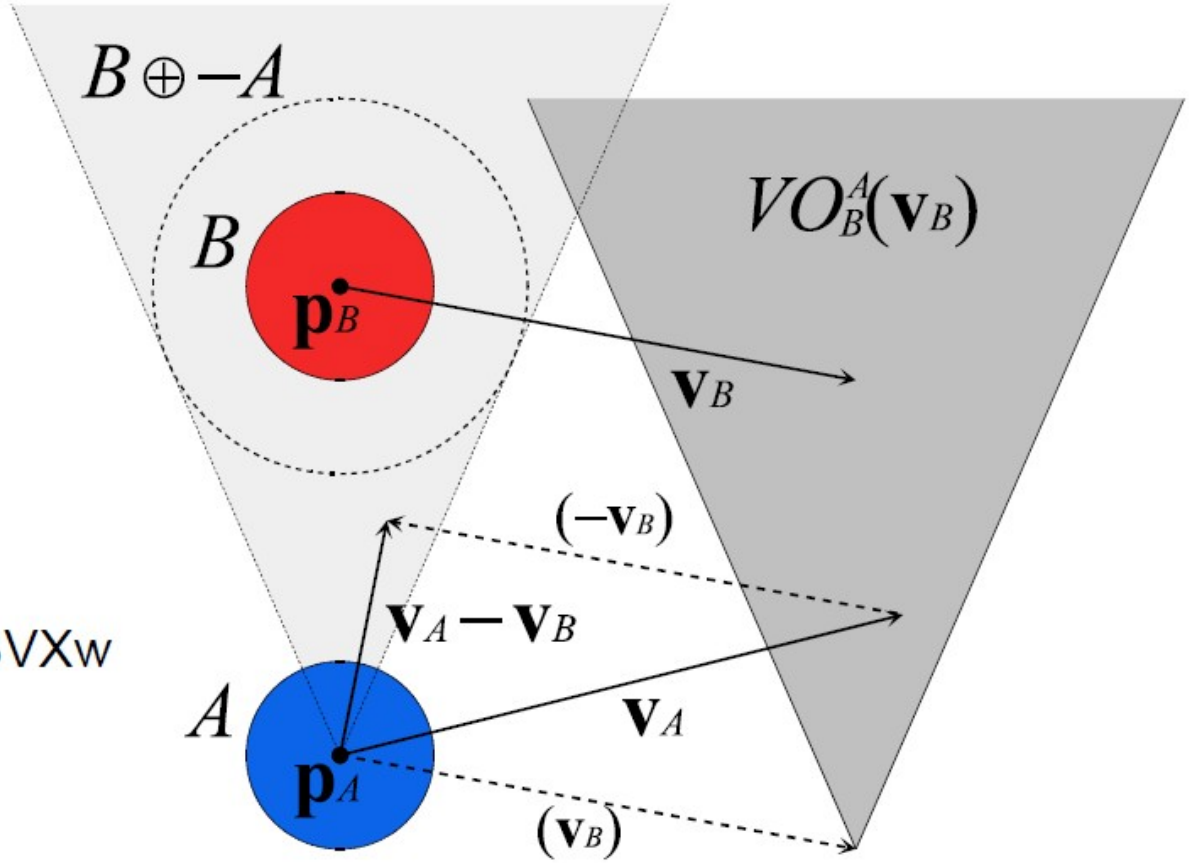


(B) Safety in turning (C) Temporary crowd (D1) Cross collision (D2) Head-on coll'n (E) Front safe area (F) Verify direction

Reciprocal Velocity Obstacle (RVO)

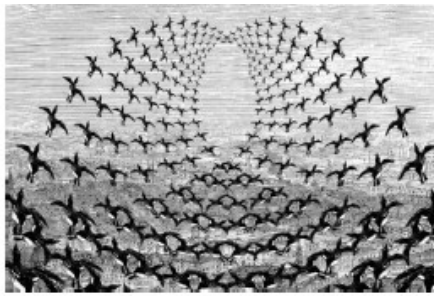


Idea: instead of choosing a new velocity outside the velocity obstacle, take the *average* of a velocity outside the velocity obstacle and the current velocity

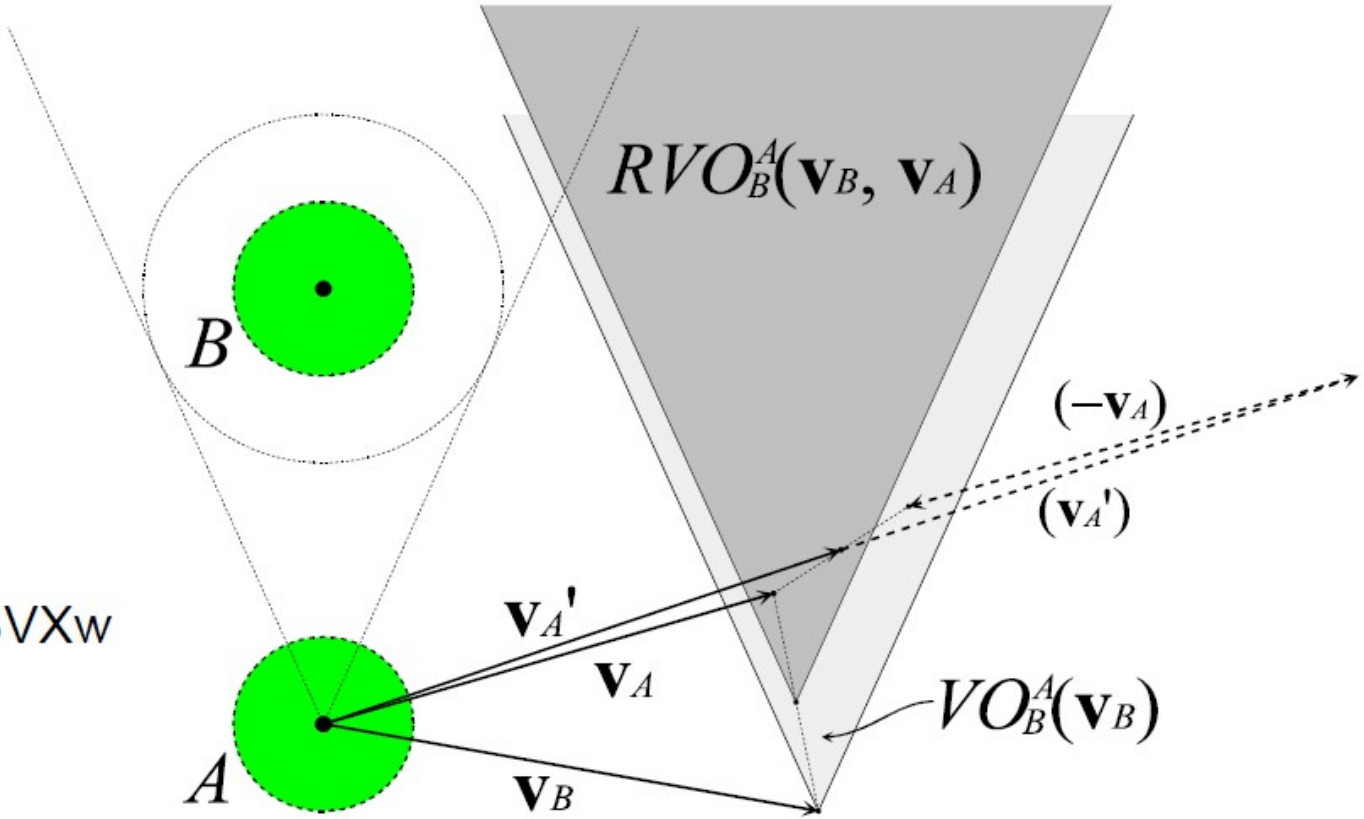


http://youtu.be/9O-YkaiBVXw

Reciprocal Velocity Obstacle (RVO)



Idea: instead of choosing a new velocity outside the velocity obstacle, take the *average* of a velocity outside the velocity obstacle and the current velocity



http://youtu.be/9O-YkaiBVXw

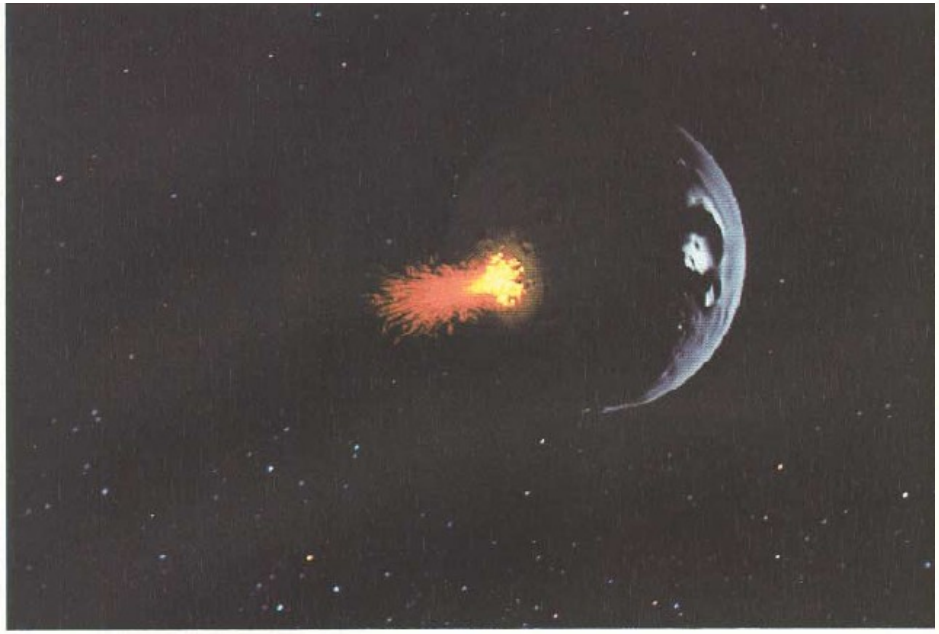
$$RVO^A_B(v_B, v_A) = \{v'_A \mid 2v'_A - v_A \in VO^A_B(v_B)\}$$

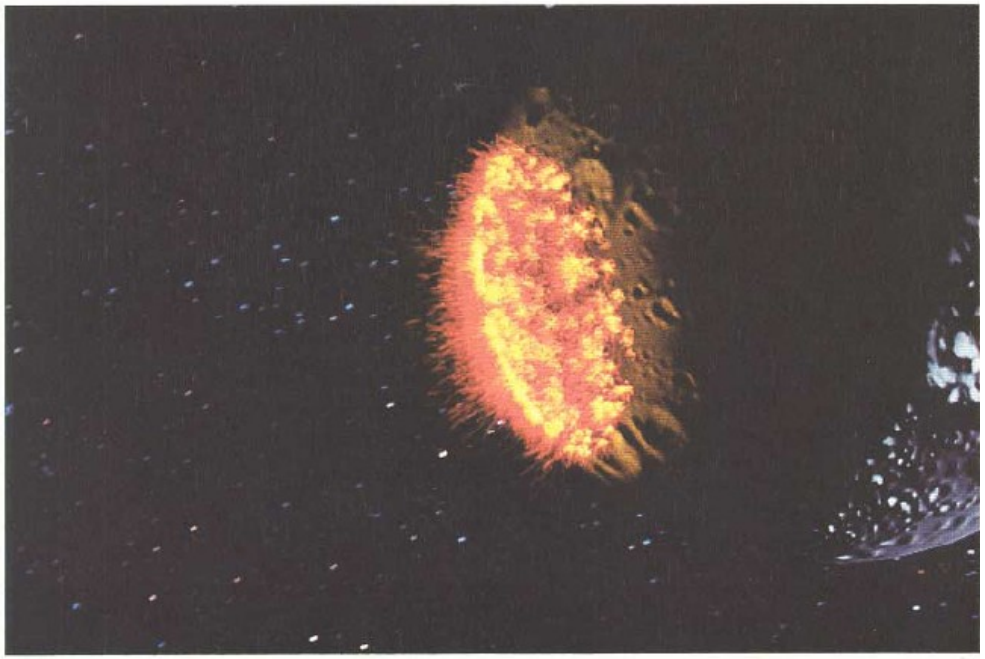
Modelling fuzzy objects with particle system

- Initially developed by Reeves to create scenes for Star Trek II.

http://www.youtube.com/watch?v=13b7TSiidaM&feature=channel_page

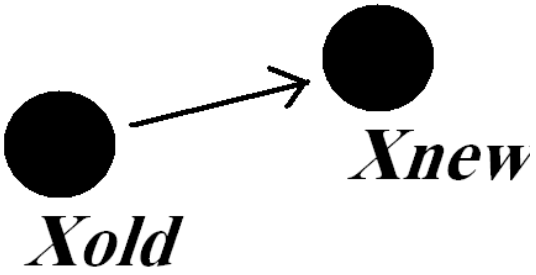
Reeves ’83,’85





Particle Dynamics

- A particle's position in each succeeding frame can be computed by its velocity
- This can be modified by an acceleration force for more complex movements, e.g., gravity simulation.



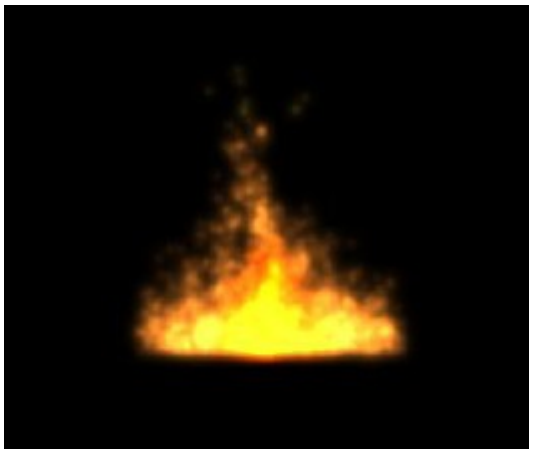
$$x_{new} = x_{old} + v_{old} \Delta t$$

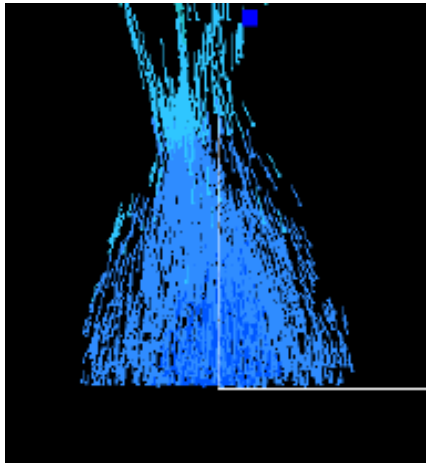
$$v_{new} = v_{old} + a \Delta t$$

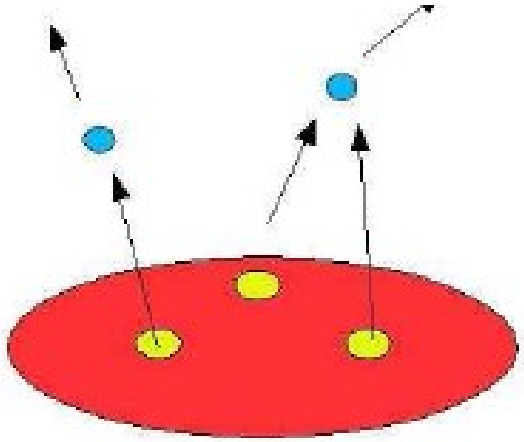
$$ma = F$$

Modelling Fire

- Define an emitter circle
- The particles are launched in a random direction within a predefined range





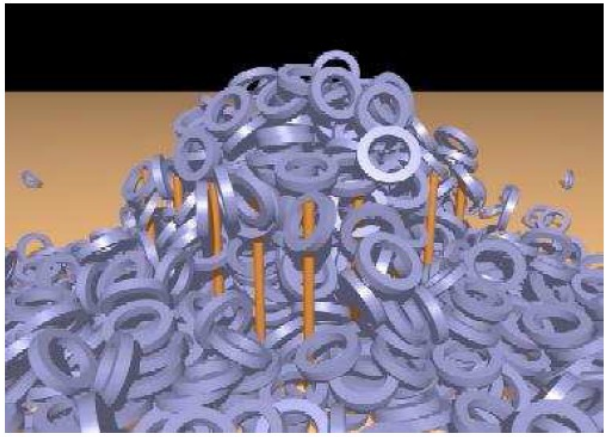


- number of particles generated :

$$N_{partsF} = N_{mean} + rand() \cdot Variance \qquad -1.0 \le rand() \le 1.0$$

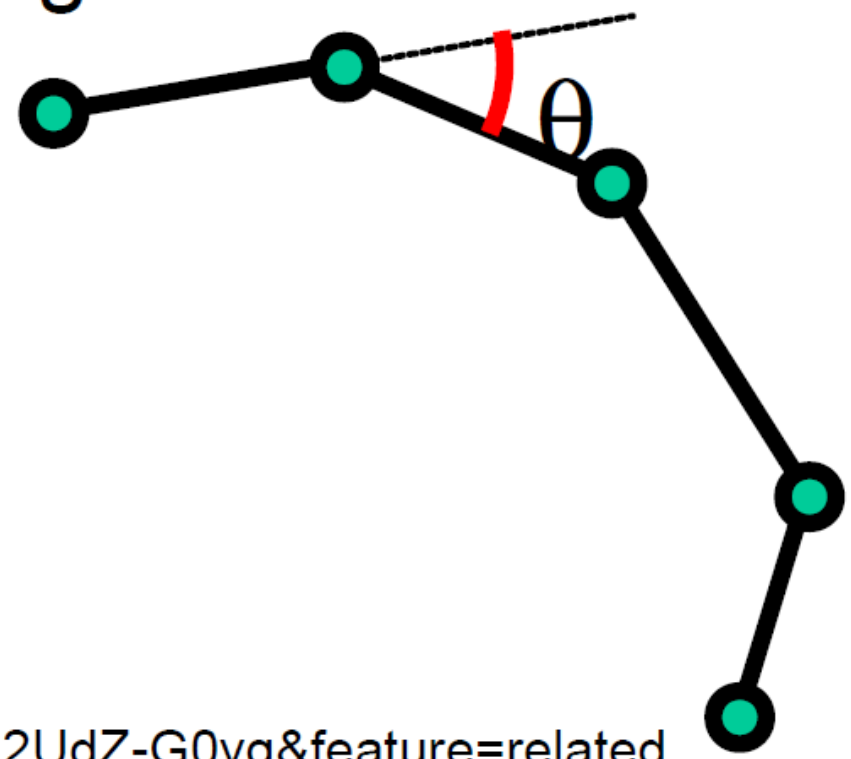
Physically-based animation

- Using Newton's laws of dynamics to simulate various phenomena
  - *Drop objects in the scene, and let them collide and see what happens*
  - *Blowing hair by a dryer*
  - *Add force to the bodies and torque to the joints and simulate the movements of human figures*



Hair

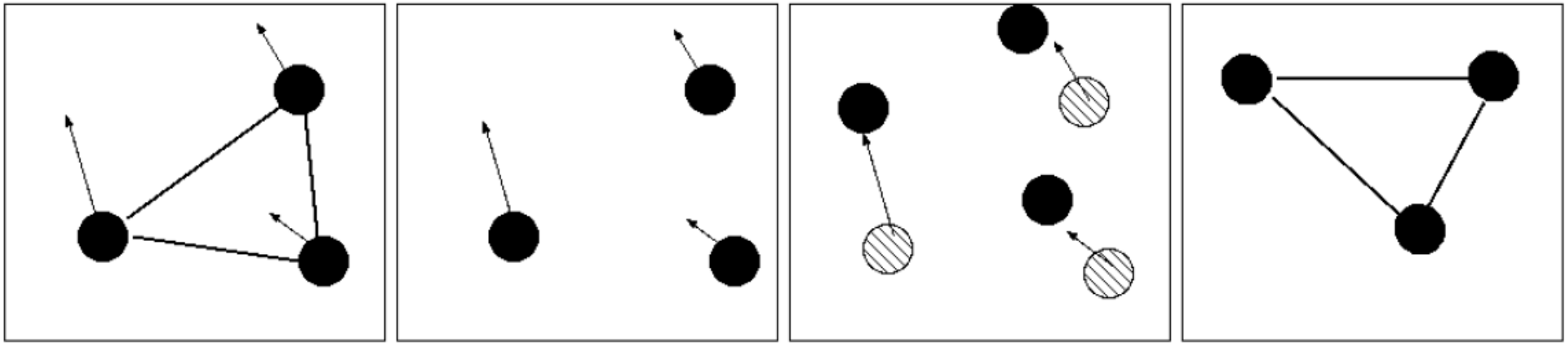
- Linear set of particles
- Length structural force
- Deformation forces proportional to the angle between segments



http://www.youtube.com/watch?v=gj2UdZ-G0vg&feature=related

ACP: How does it work?

- Model rigid bodies by a collection of particles
- Assume the distance between the particles will remain the same if they belong to the rigid body
- Each particle will move based on the free particle dynamics
- The 3D position will be modified according to the distance constraints



ACP: Modeling Characters

- We can model human characters by this system
- Used for ragdoll physics in Hitman
- http://www.teknikus.dk/tj/gdc2001.htm
- Can apply for inverse kinematics

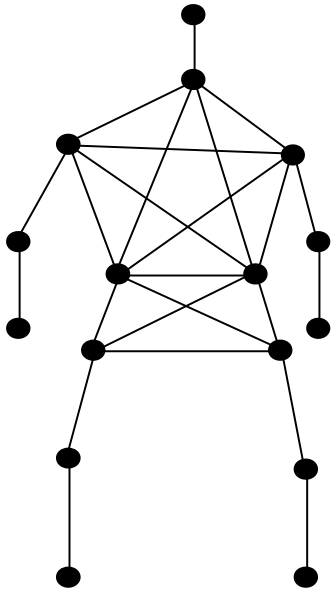
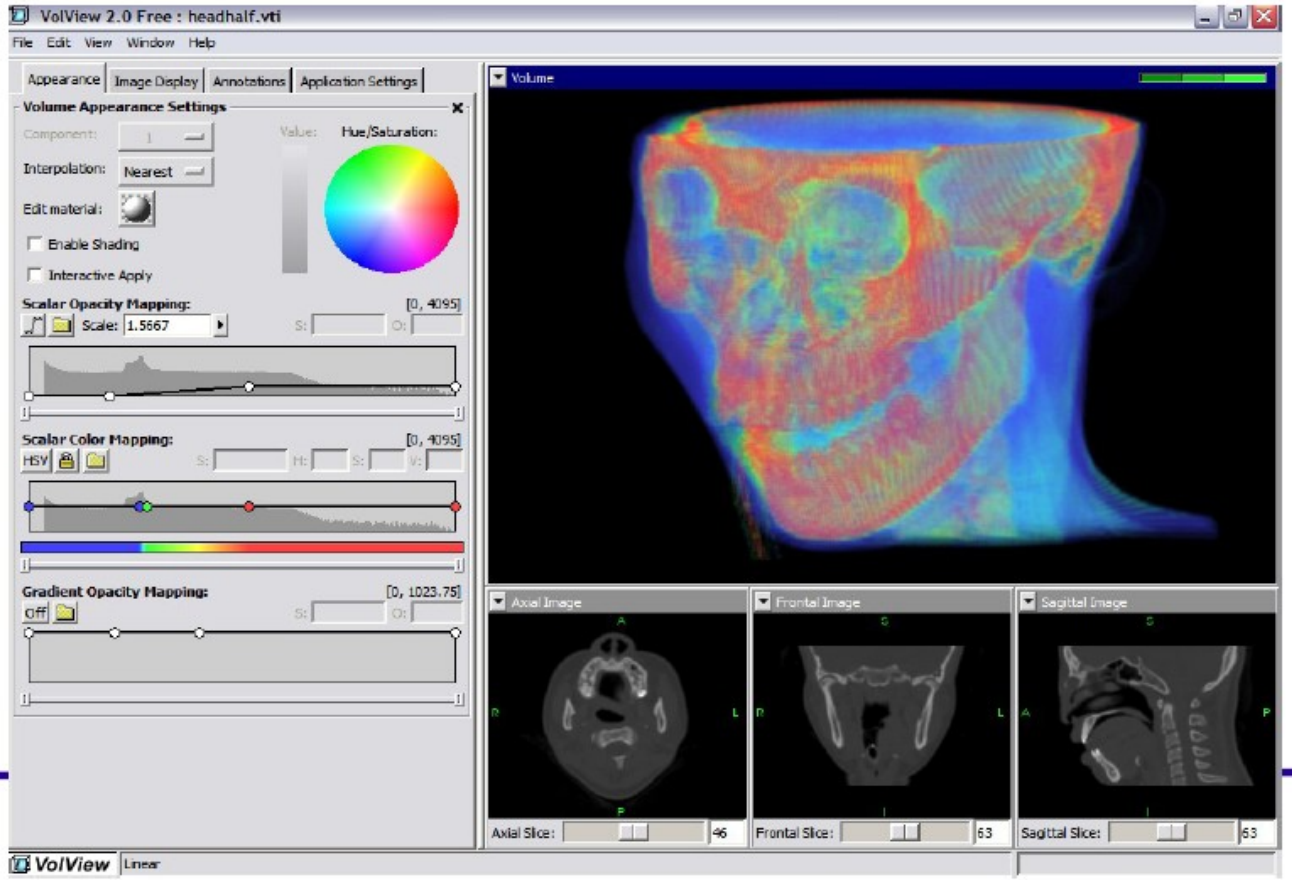


Figure 9. The particle/stick configuration used in Hitman for representing the human anatomy

Volume Rendering

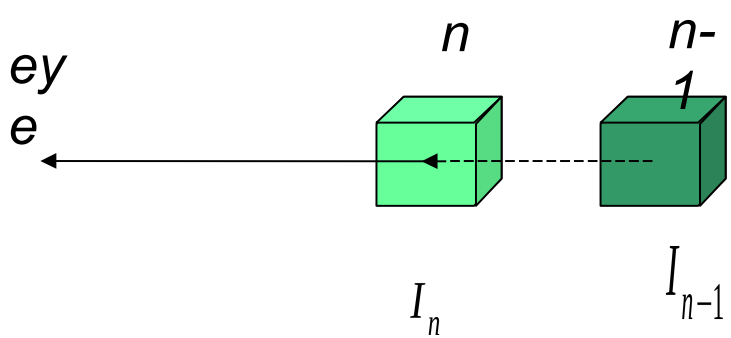
- Display the **information inside the volume using colours and transparency**
    - **(Defining a colour / opacity transfer functions)**
- direct display : volumetric data → screen pixels



Rendering with Transparency

- Back to front ray casting
  - Only need to store current value of I

$$I_n = A_n E_n + (1 - A_n) I_{n-1}$$

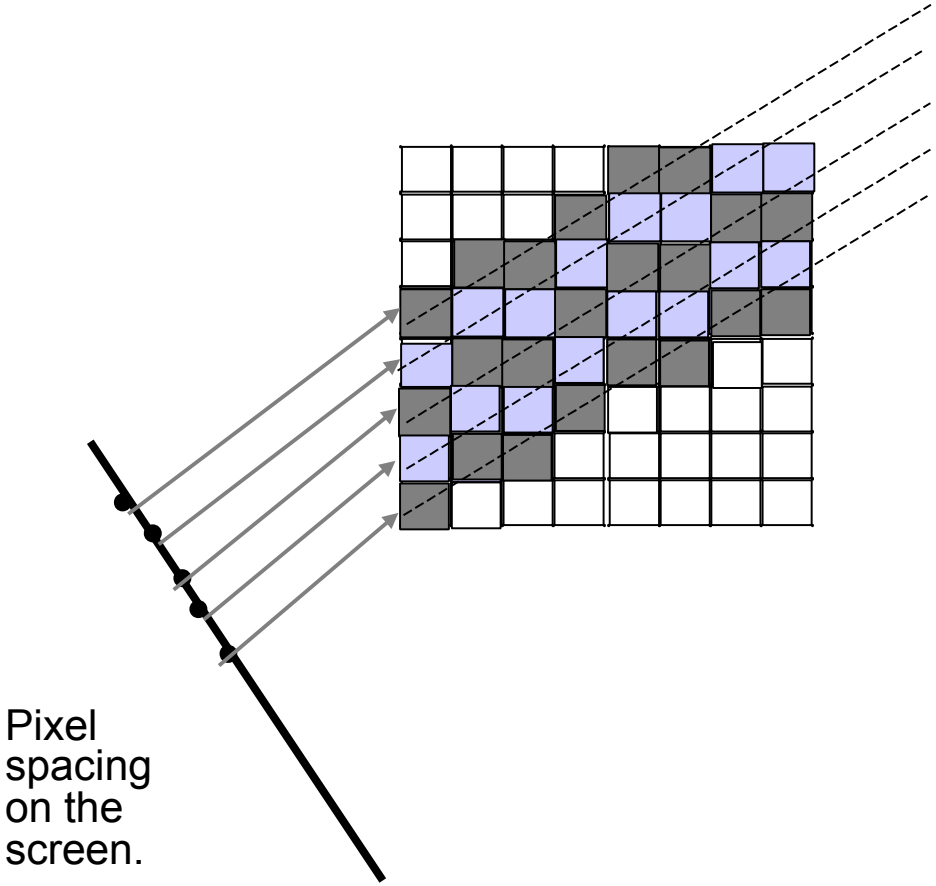


Subscript *n* refers to cell n.

A refers to object *opacity.*

- Start with furthest away cell and blend towards the camera.
- I_n corresponds to current contents of the frame buffer.
- E_n Light emitted from cell n

Voxel Traversal

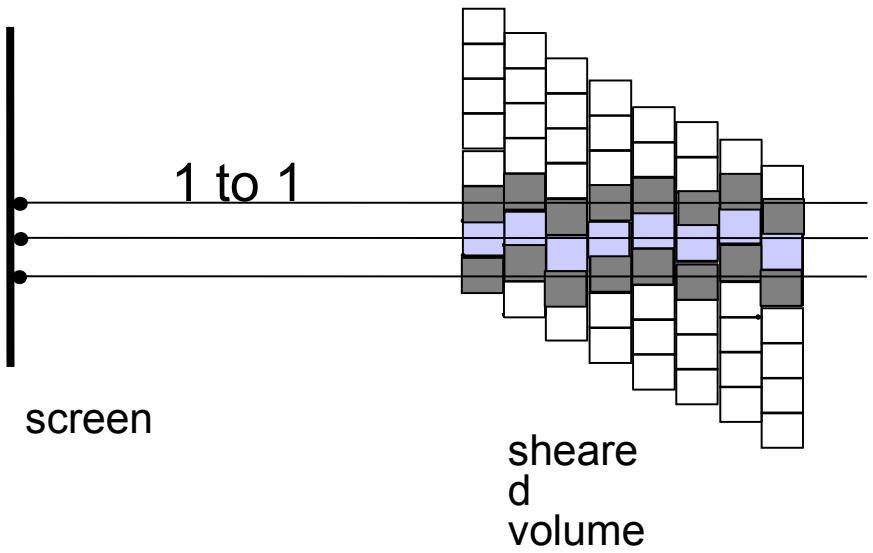


If we originate rays in each voxel at the same position of each grid, each ray forms an identical path of voxels through the grid

- Path known as a **template**
- All identical, pre-compute, save computation

Shear-warp factorisation

- Instead of traversing along rays, visit voxels in a plane
  - regardless of camera viewing angle [Lacroute '94]
- Perform final warp on the image due to the shear process
- Assumes a orthographic camera model
  - projection perpendicular to image plane

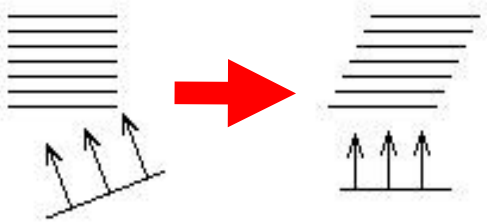


· Rays are cast from the base plane voxels at the same place.

· They **intersect voxels on subsequent planes in the same location.**
The weights (the contribution of each voxel) are easy to compute

· Only **one set of interpolation weights needs to be computed for all the voxels in a slice**

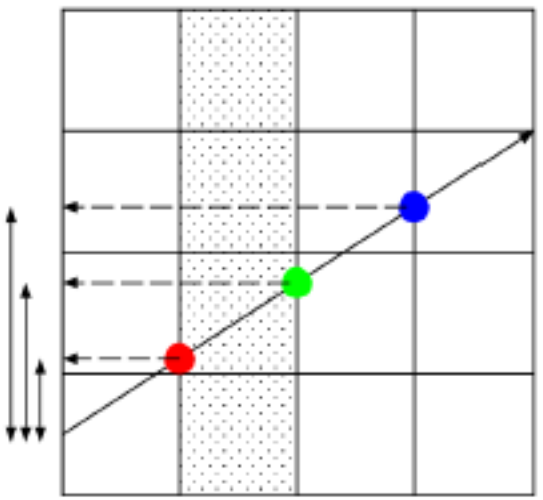
- **volume is sheared** so that rays remain perpendicular to base plane
  - lead to **efficient ray computation**

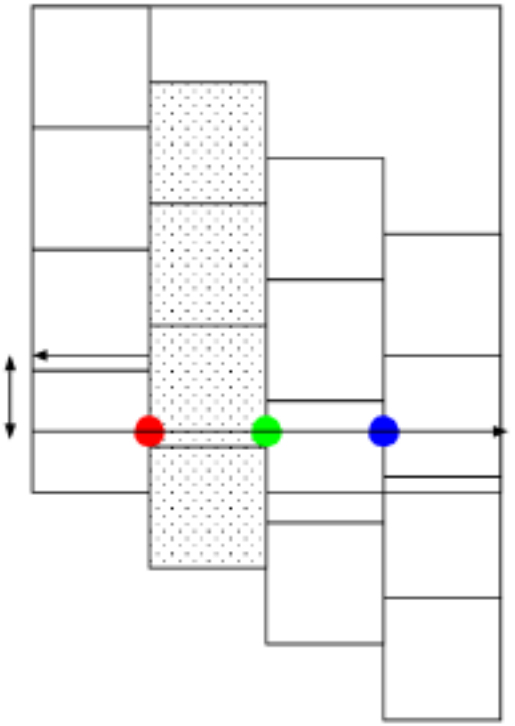


How to decide how much to shear?

Calculate the intersection of the ray and the front plane

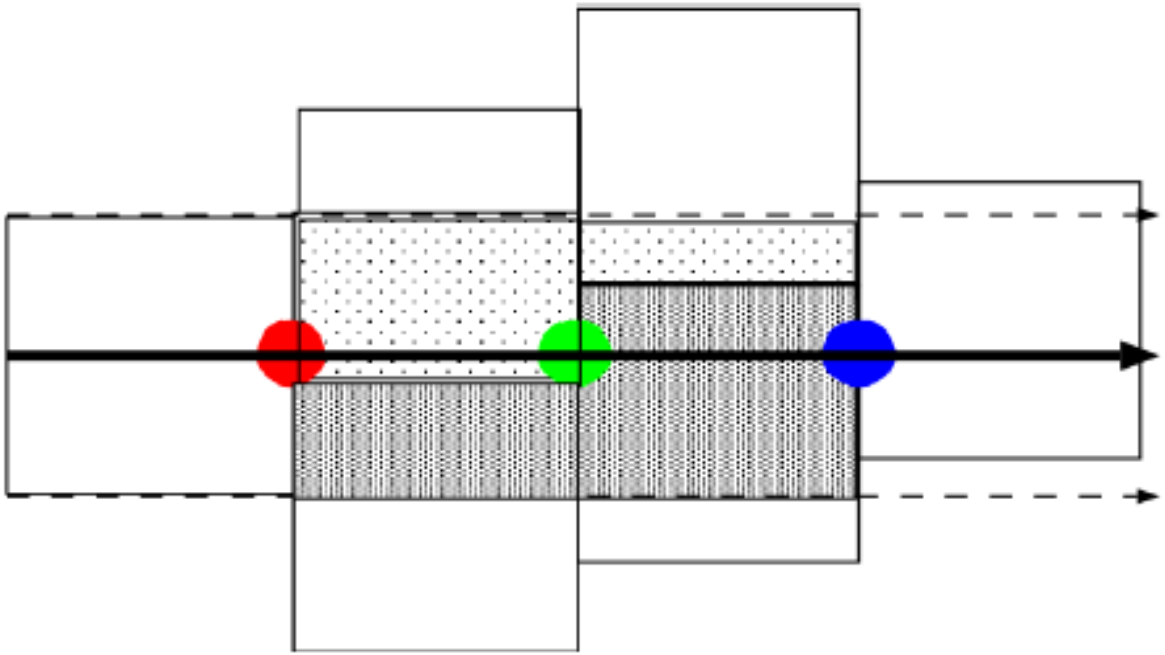
That gives the offset





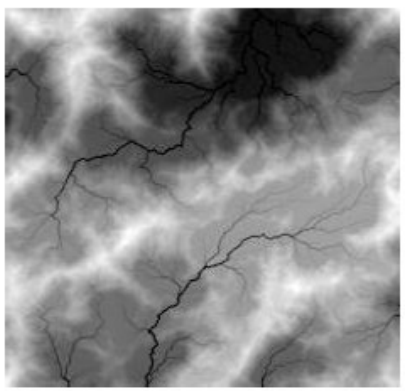
How to decide weights?

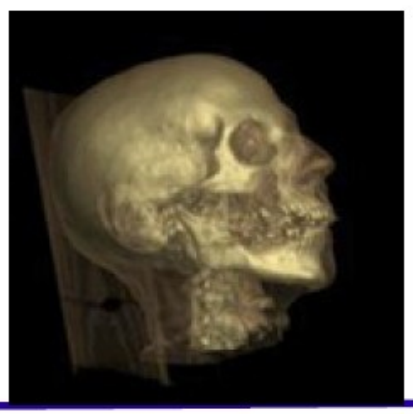
Can compute it based on how much the pixel is overlapped with the voxel

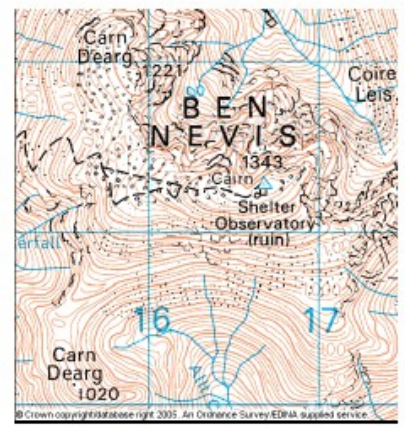


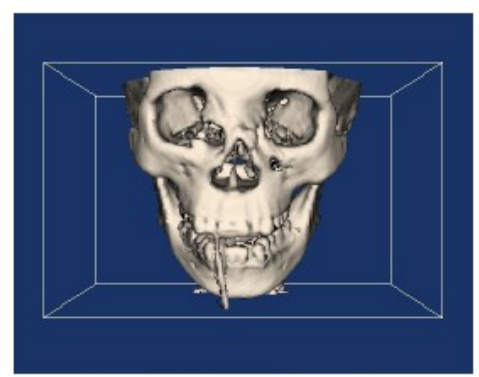
Contouring

- Input : 2D or 3D grid with scalar values at the nodes
- Output : Contours (polylines, polygons) that connect the vertices with the same scalar value

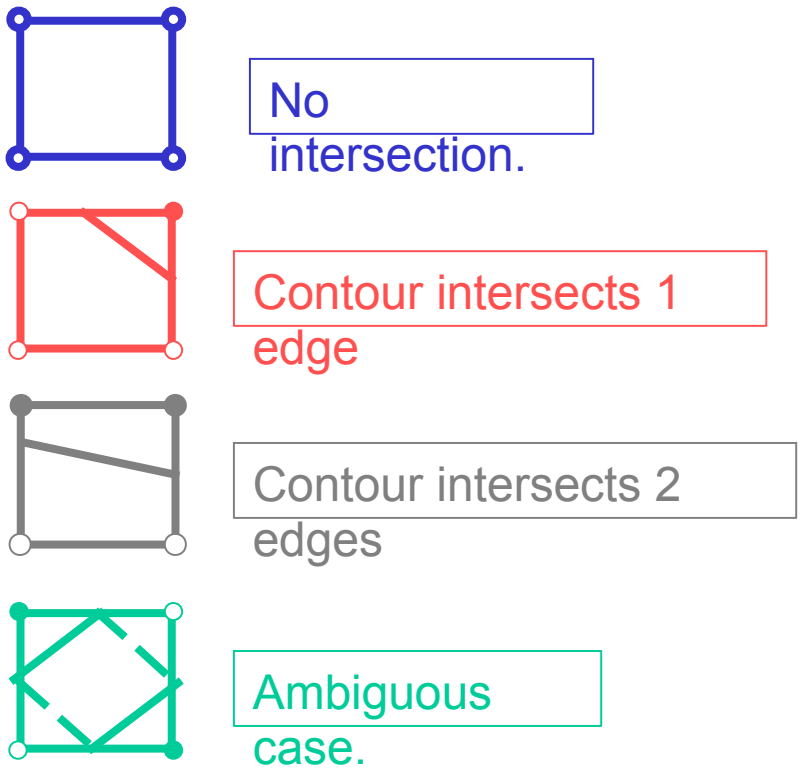


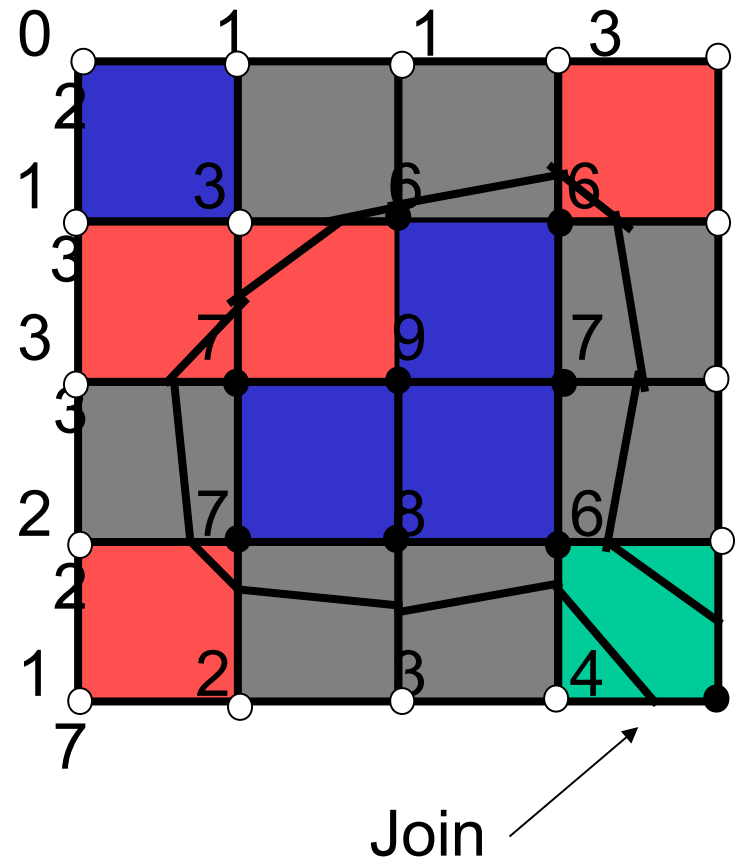






Marching Squares





- Finally : resolve any ambiguity
  - here choosing “join” (example only)

Marching Cubes

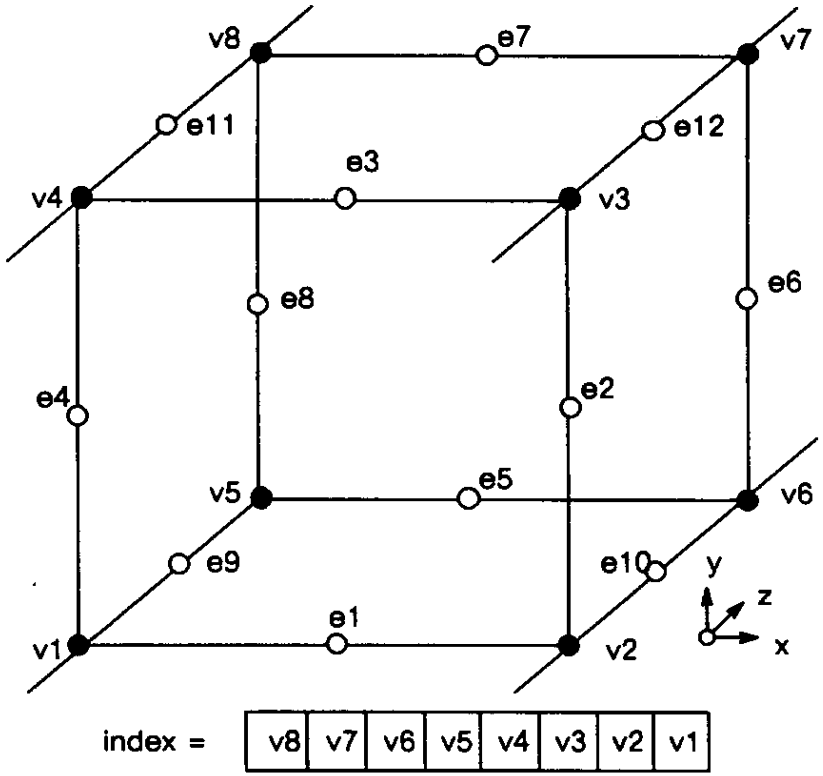


Figure 4. Cube Numbering.

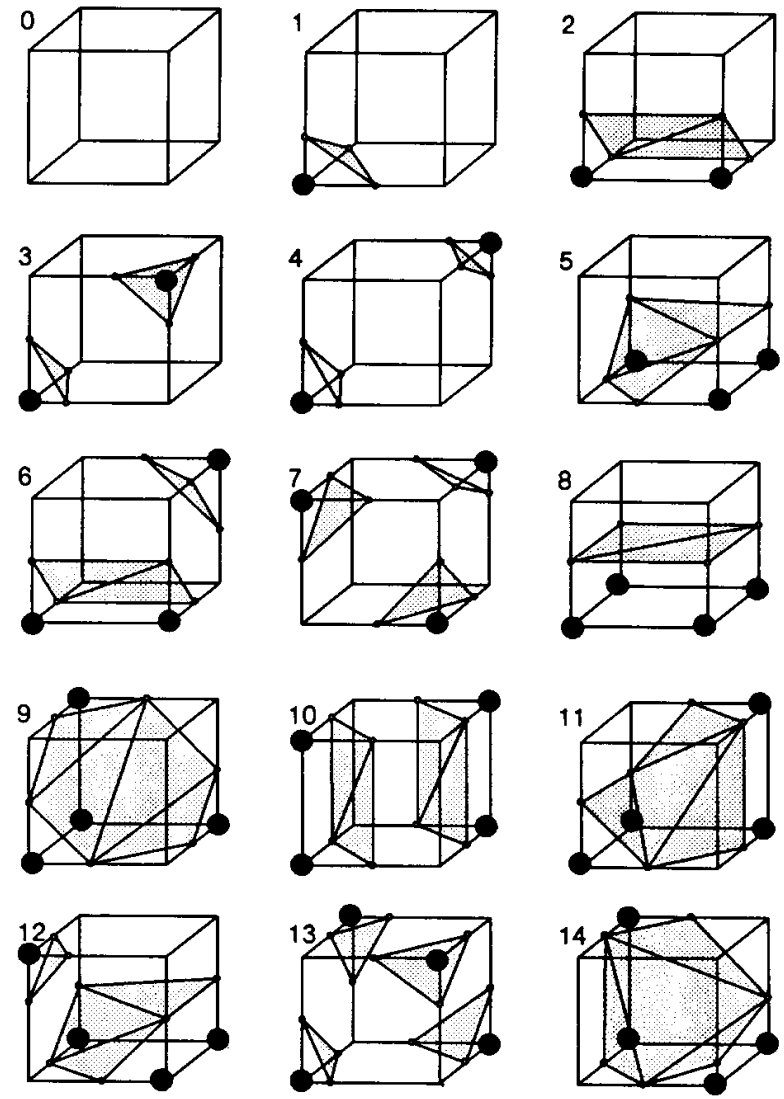


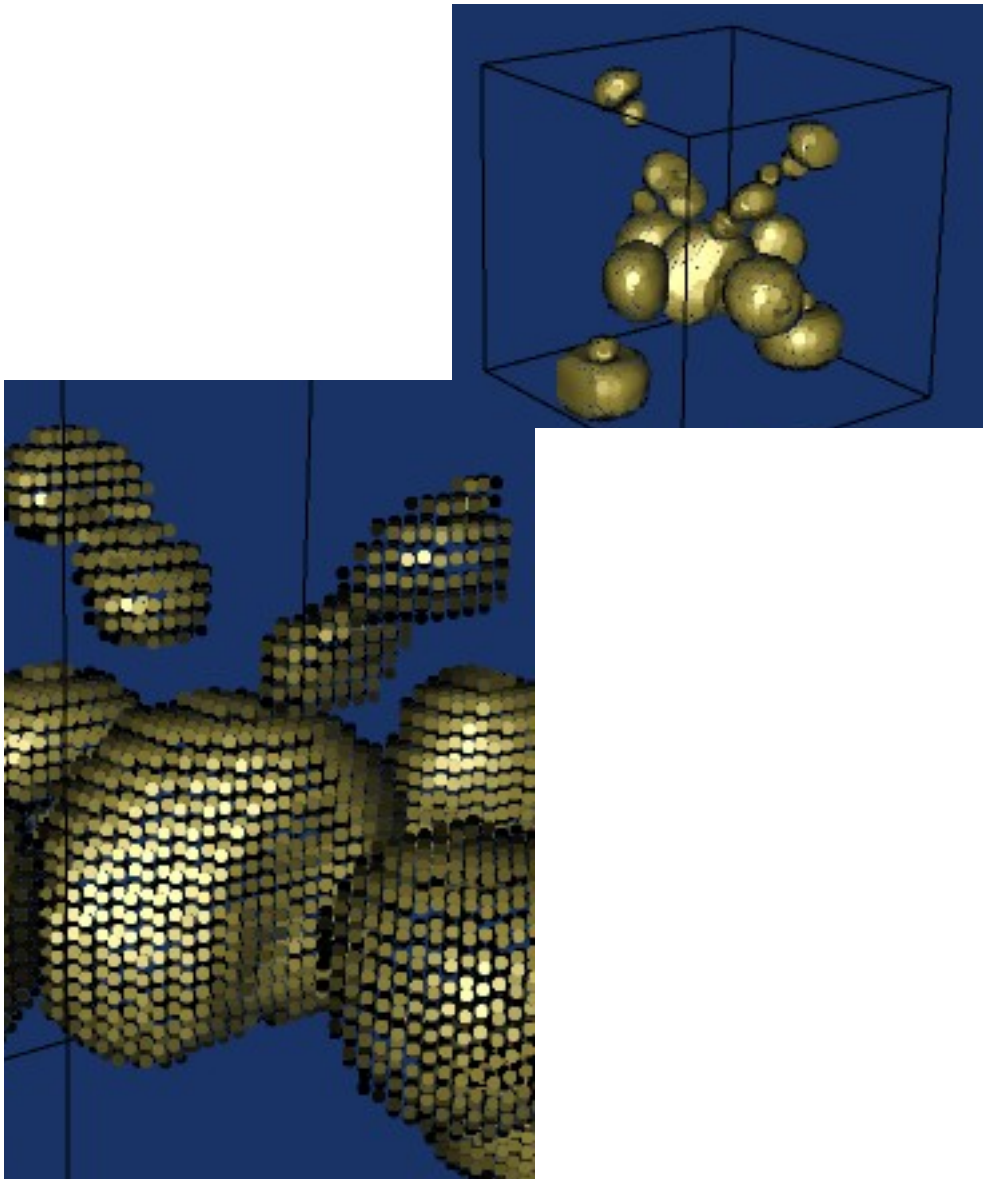
Figure 3. Triangulated Cubes.

- Ambiguous cases
  - 3,6,10,12,13 – split or join ?

Dividing Cubes : Example

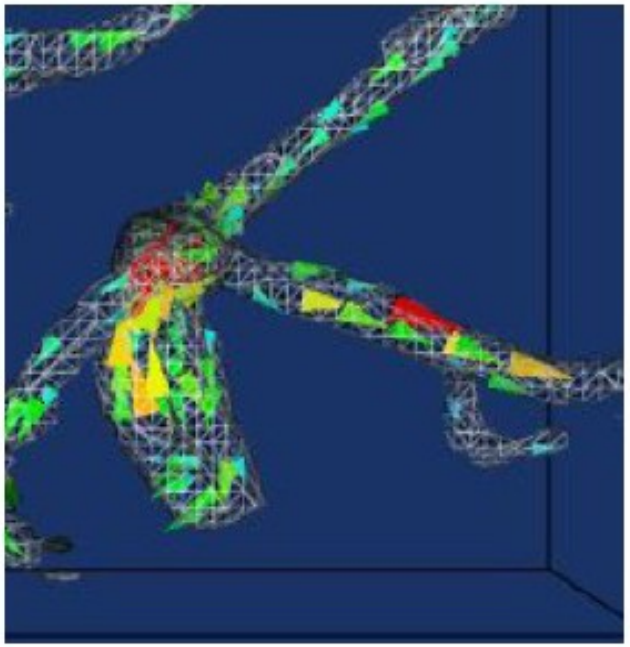
50,000 points

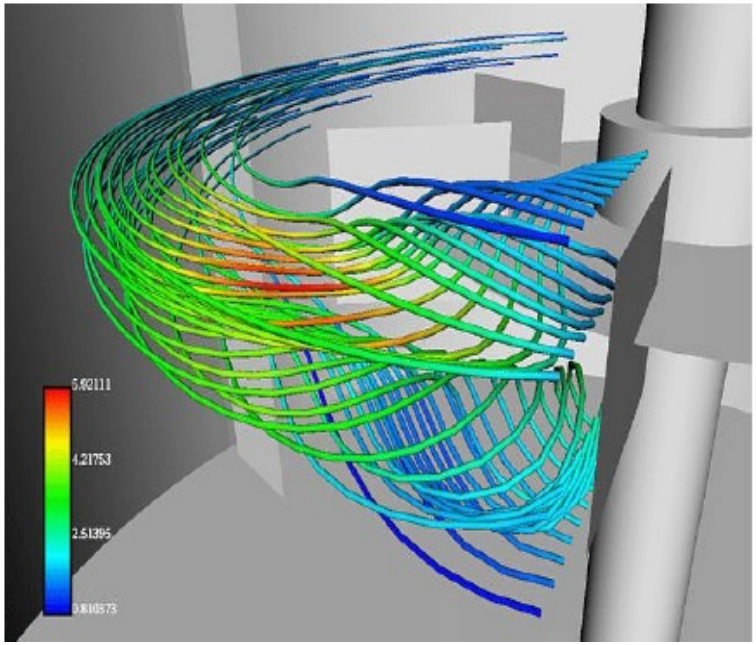
when sampling less than screen resolution structure of surface can be seen



Flow Visualisation

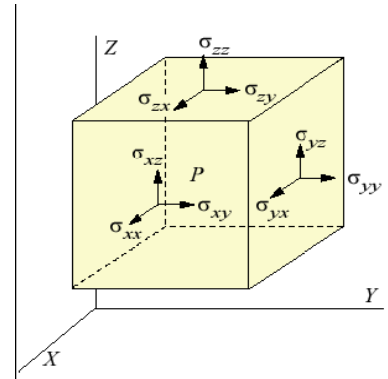
- Glyphs, streamlines, streaklines





Tensor Visualisation

- **Stress/strain tensors**
  - analysis in engineering

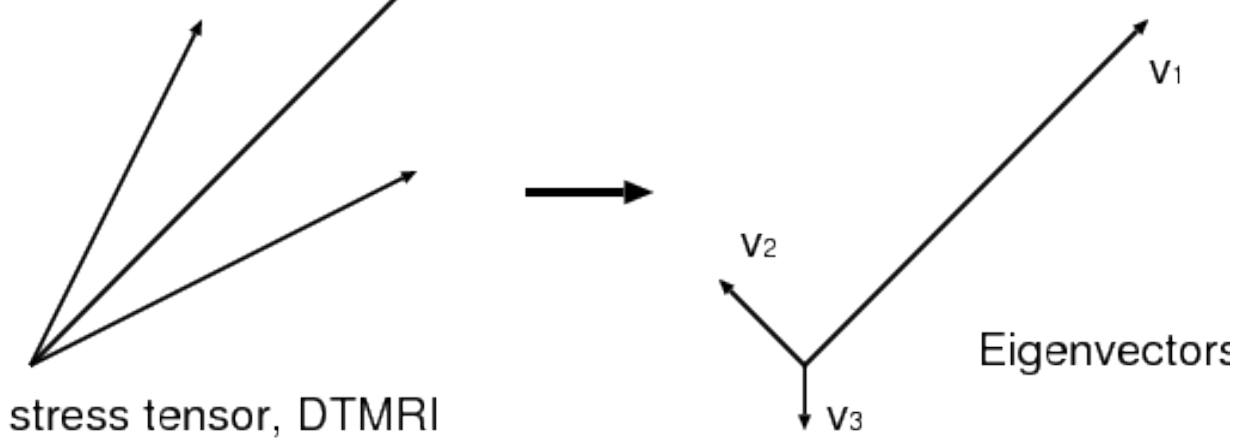


- **DT- MRI**
  - molecular diffusion measurements

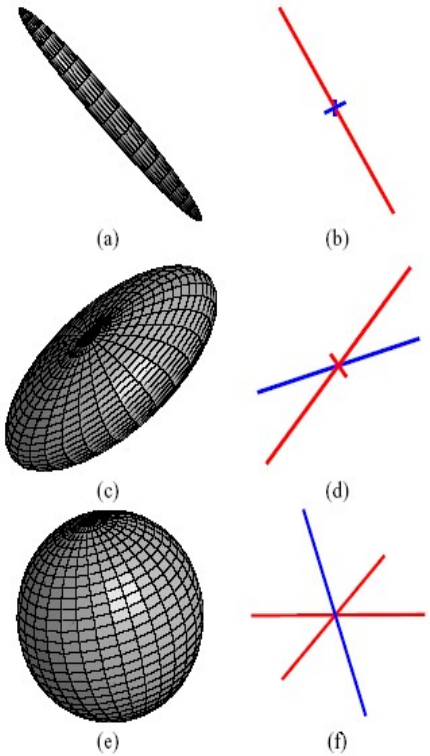
- **These are represented by 3x3 matrices**
  - *Or three normalized eigenvectors and three corresponding eigenvalues*

Computing Eigenvectors

- 3x3 matrix results in **Eigenvalues** (scale) of normal stress along **eigenvectors** (direction)
- Form 3D coordinate system (locally) with mutually perpendicular axes
- Ordering by eigenvector referred to as **major, medium and minor eigenvectors**

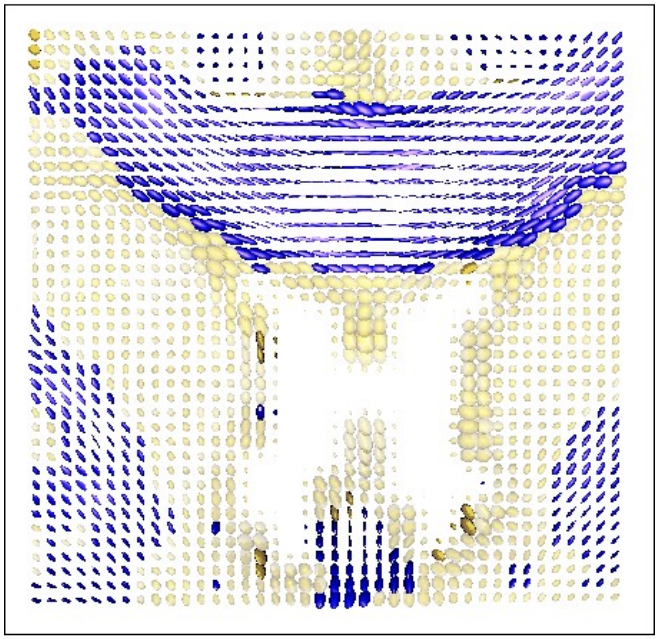


Tensor Glyphs



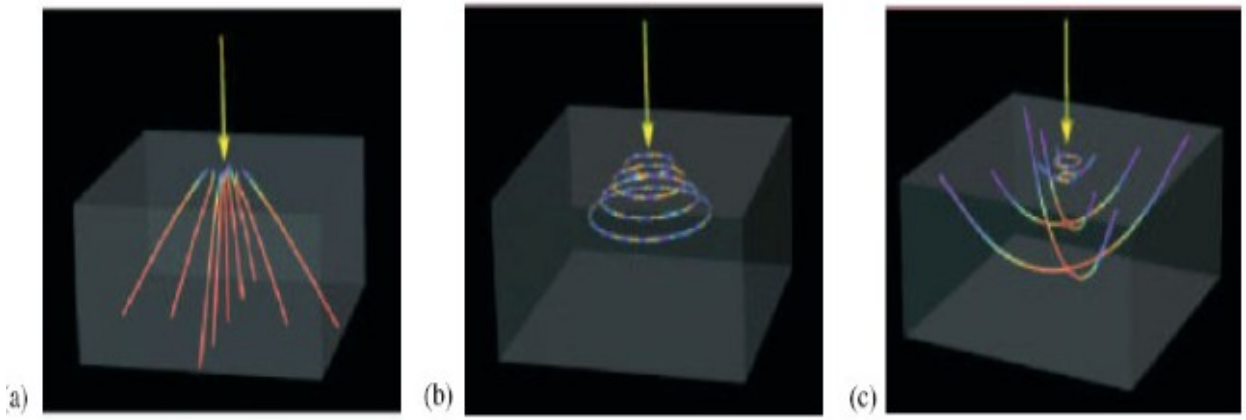
- **Ellipsoids**
  - **rotated into coordinate**

  system defined by eigenvectors of tensor
  - axes are **scaled by the eigenvalues**
  - **very suitable as 3 modes of variation**
- **Classes of tensor:**
  - (a,b) - **large major eigenvalue**
    - ellipse approximates a **line**
  - (c,d) - **large major and medium eigenvalue**
    - ellipse approximates a **plane**
  - (e,f) - **all similar** - ellipse approximates a **sphere**



Various techniques for tensor visualisation

- Streamlines



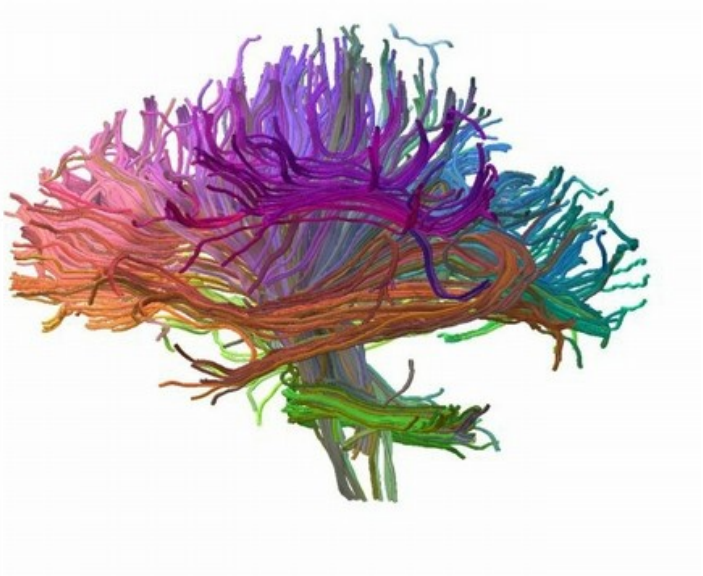
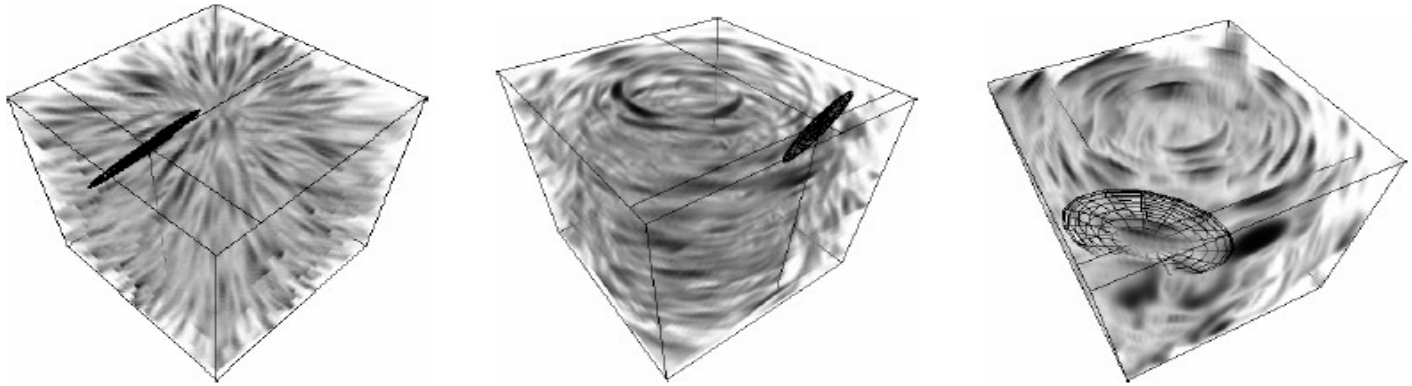


Figure 8. Hyperstreamlines for minor, intermediate and major principal stress for a point-load.

LIC

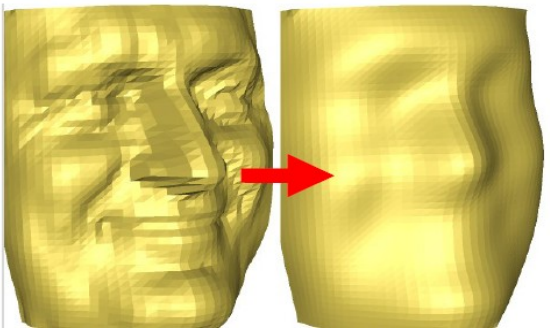


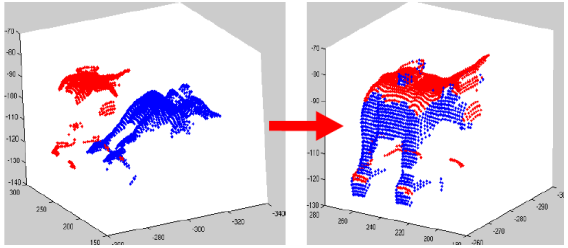
(a) (b) (c)

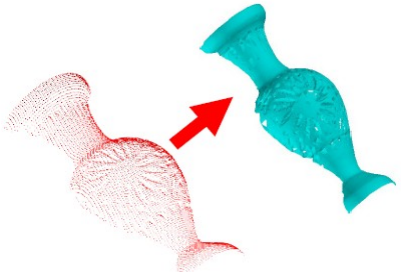
u.se/

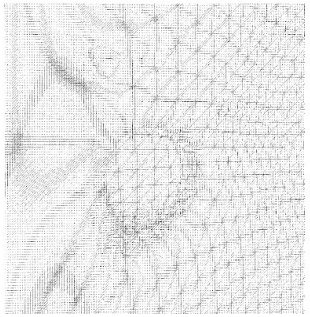
Processing 3D Surface Data

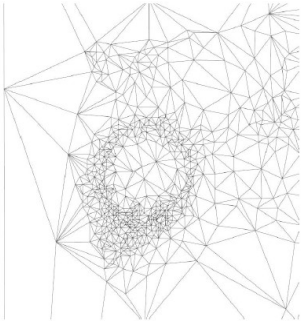
- 1) Capture the data (by stereo vision, range scanner)
- 2) Registration (if the data was captured by multiple attempts)
- 3) Adding the topology : converting to mesh data
- 4) Smoothing
- 5) Decimation
- 6) Remeshing











Metaballs (implicit surface)

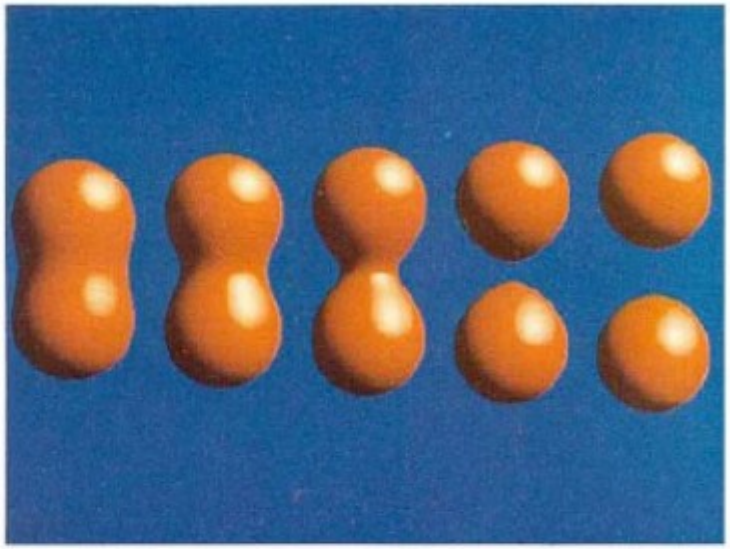
- Create an implicit function similar to the electron density maps

$$D(x,y,z)=\sum_i b_i \exp(-a_i r_i)$$

$$r_i=\sqrt{(x-x_i)^2+(y-y_i)^2+(z-z_i)^2}$$

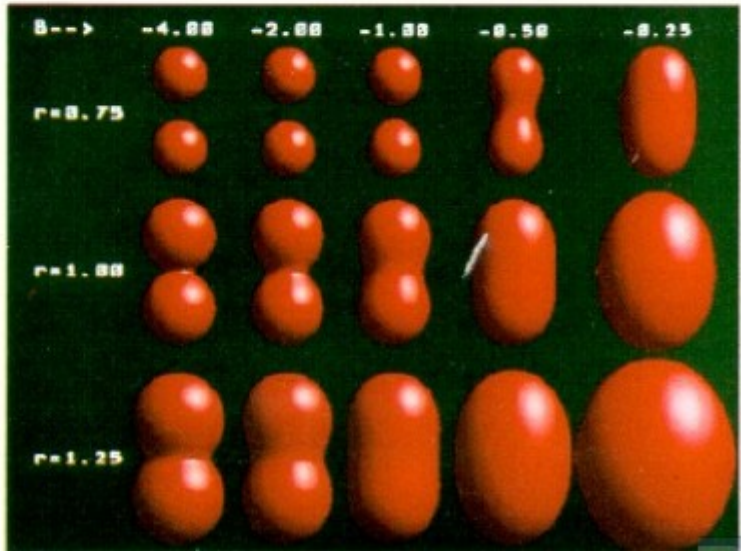
(x_i, y_i, z_i): the center of the particle i

$$F(x,y,z)=D(x,y,z)-T$$



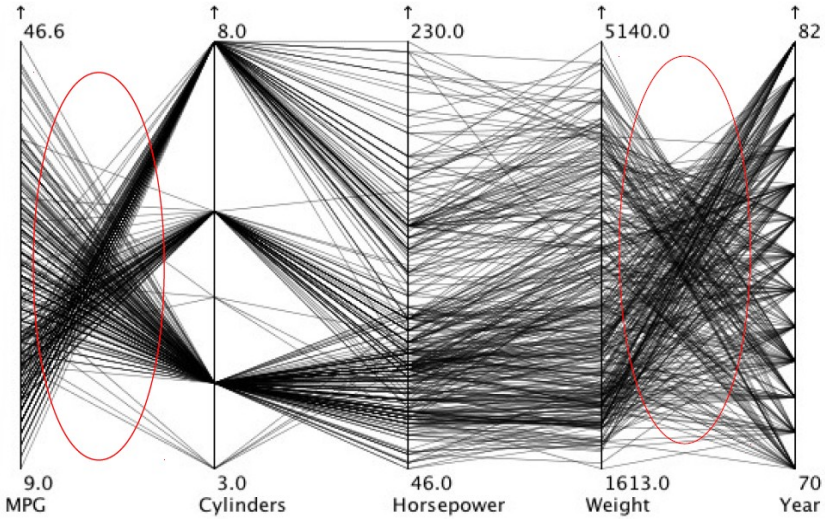
- We can define the surface where $F(x,y,z)=0$

  Blinn '92



Information Visualisation

- Parallel coordinates – multivariate data
- Graph visualisation
- Literature visualisation



Thank you !!

- Good luck