Games Engine Design

Course SS 2015

Rüdiger Westermann
Lehrstuhl für Computer Graphik und Visualisierung

Object representation

- The geometric game objects are typically created via so-called modelling systems

- Modelling can be separated into two parts:

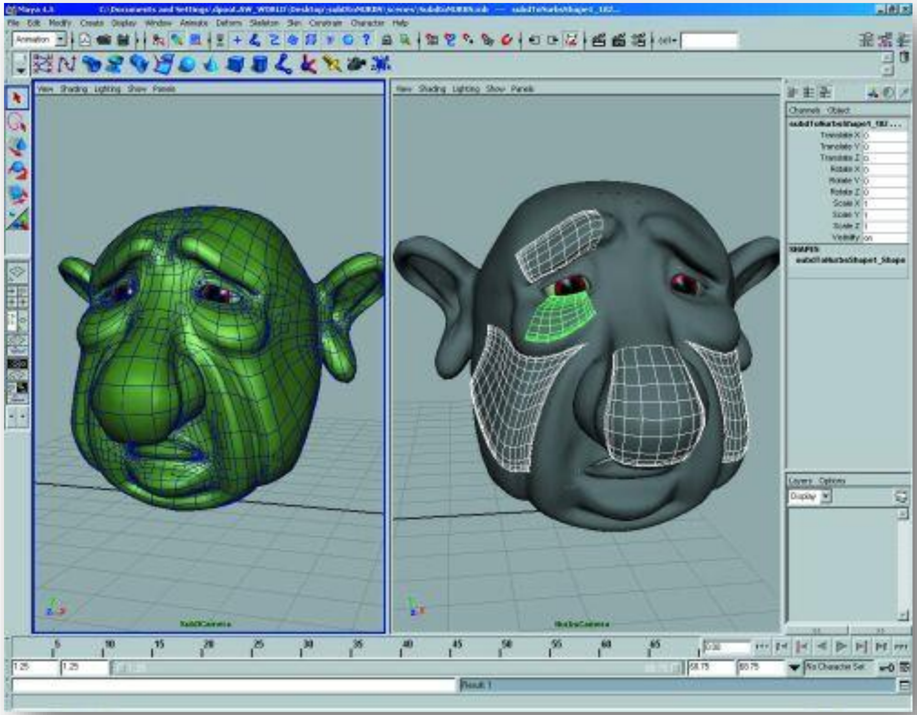
  Shape (geometric) modelling: place geometric primitives such that they represent your model well

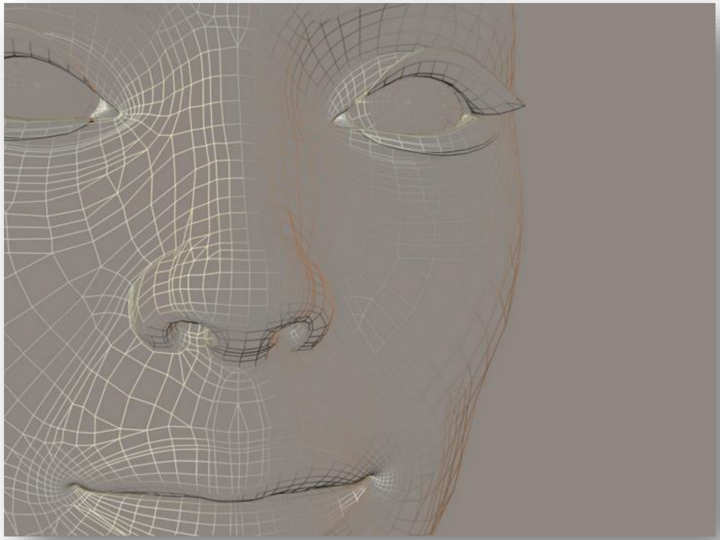
  Appearance modelling: assign material properties like color, reflection properties, textures* etc. to the object

* http://de.wikipedia.org/wiki/Textur_(Computergrafik)

Geometric model representation

- In games we typically use polygonal approximations of continuous objects



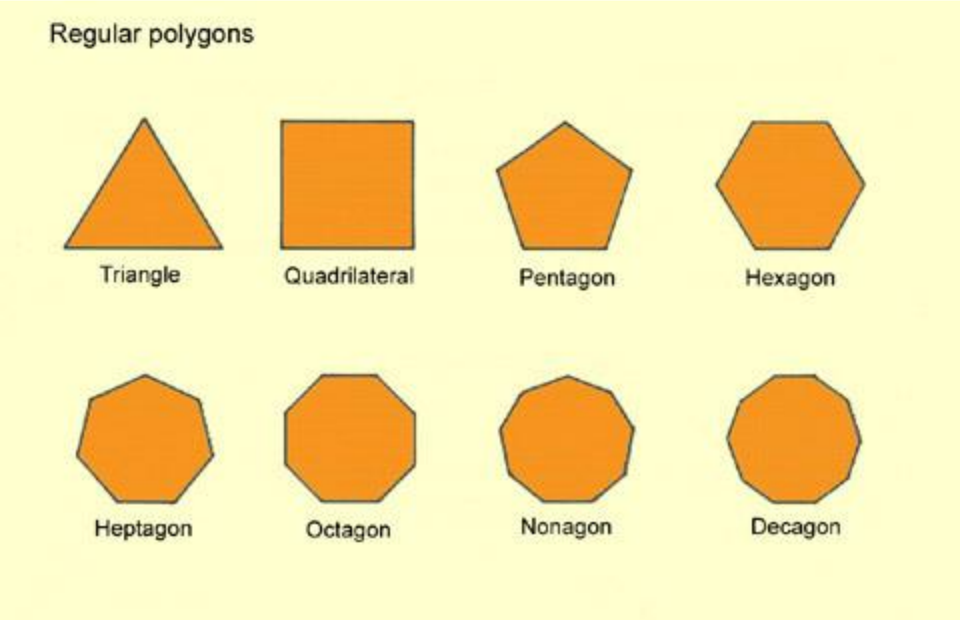


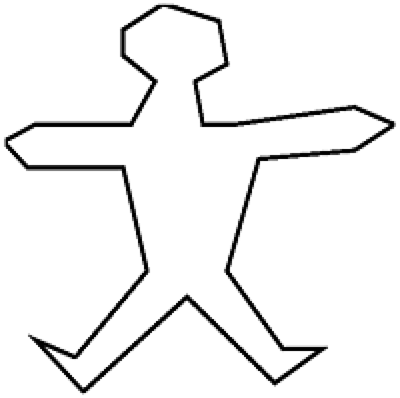
Geometric model representation

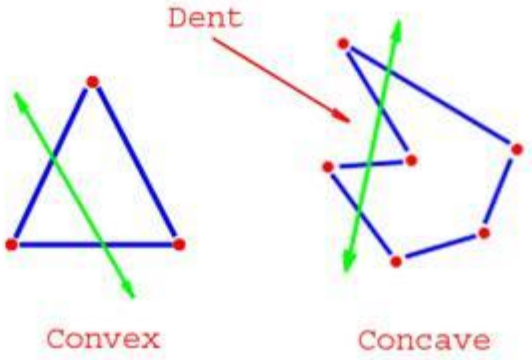
- What is a polygon?
  - It consists of points, typically in 3D-space $\mathbb{R}^3$
    - A point is represented by its x/y/z-coordinates; it is called a vertex
  - Points are connected via line segments (edges)
    - Line segments are specified in terms of the vertices at their endpoints
  - A polygon is the interior of a closed planar connected series of line segments
    - The edges do not cross each other and exactly two edges meet at every vertex

Geometric model representation

- What is a polygon?

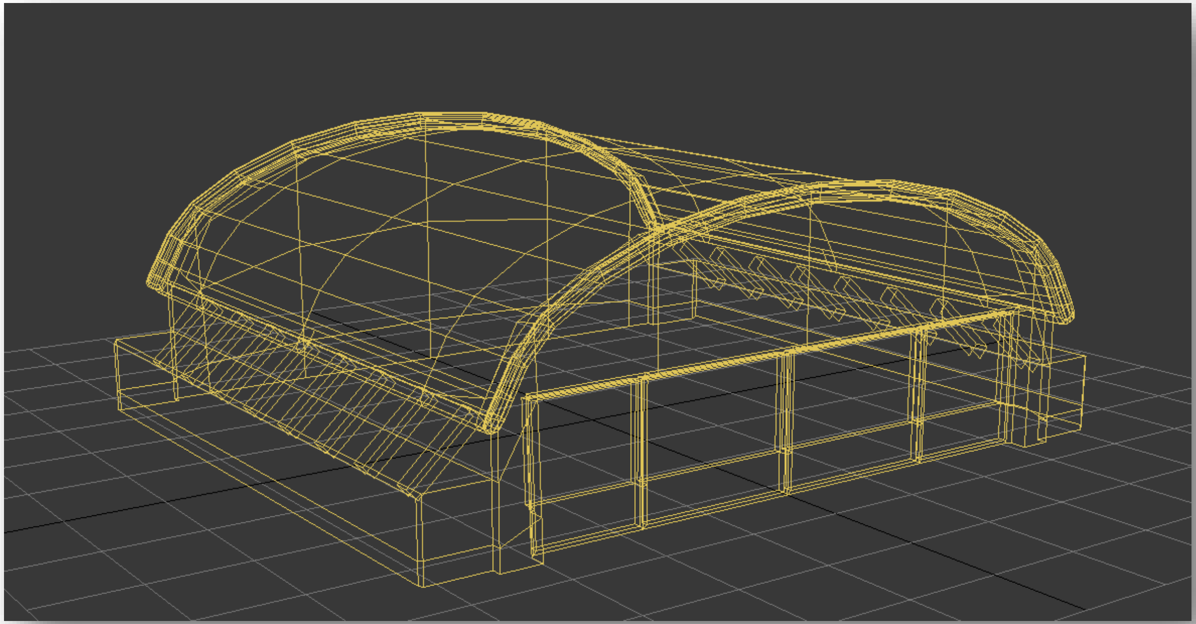






Geometric model representation

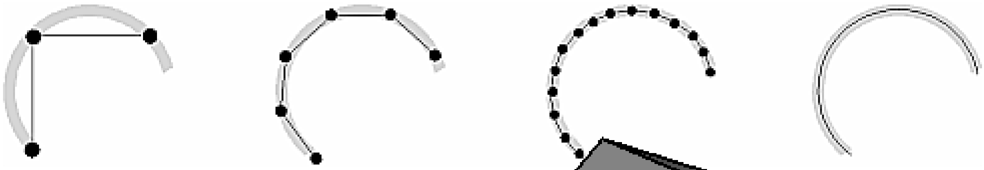
- What is a polygon mesh?
  - A surface composed of polygons (faces)
  - Objects are assumed to be hollow

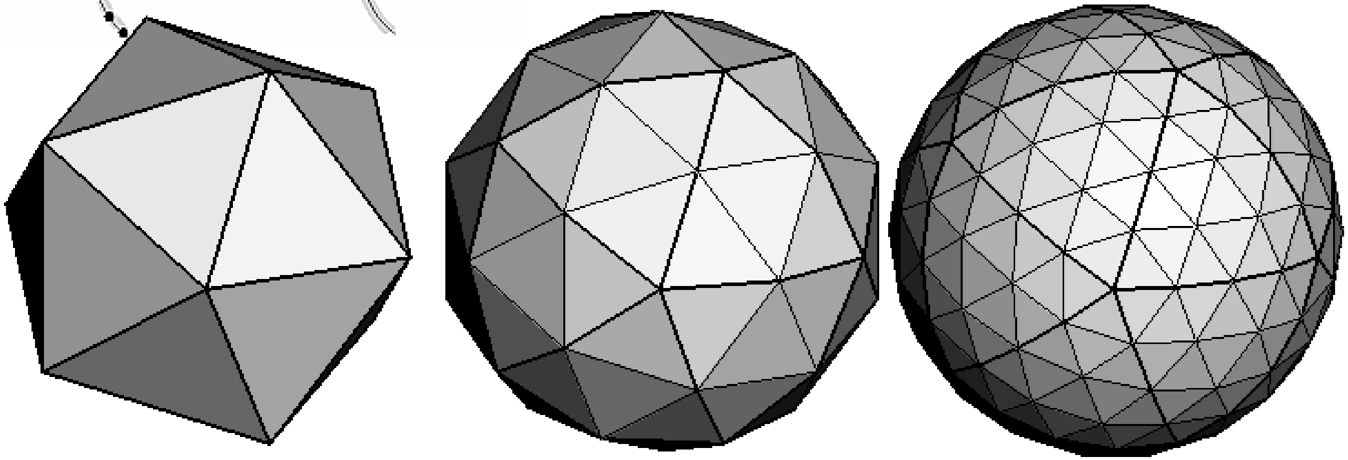


Geometric model representation

- A polygonal approximation is an approximation of a continuous surface by a polygon mesh

Approximation of a curve

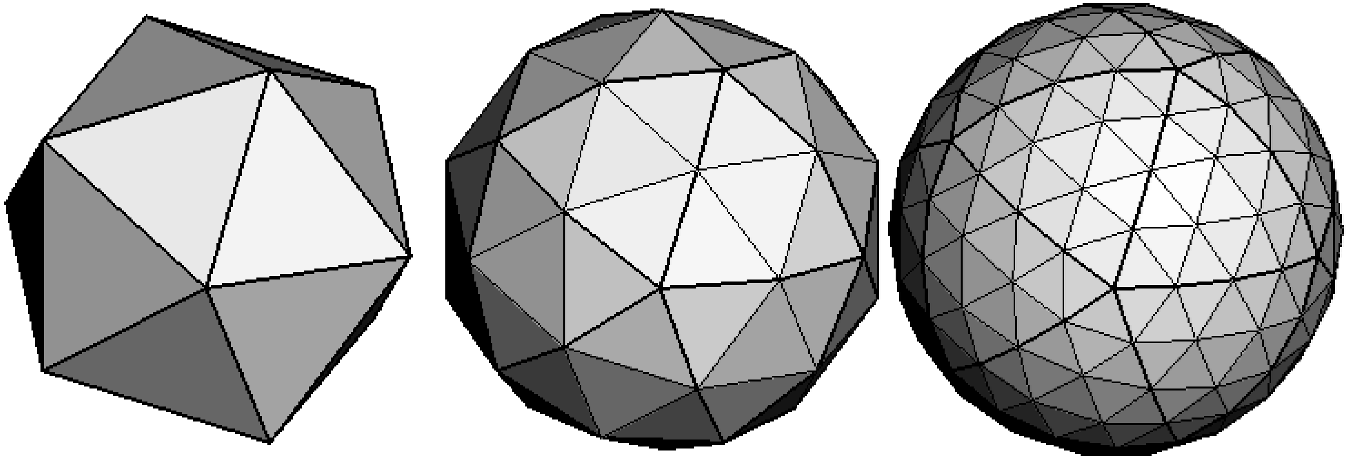




Different approximations of a sphere

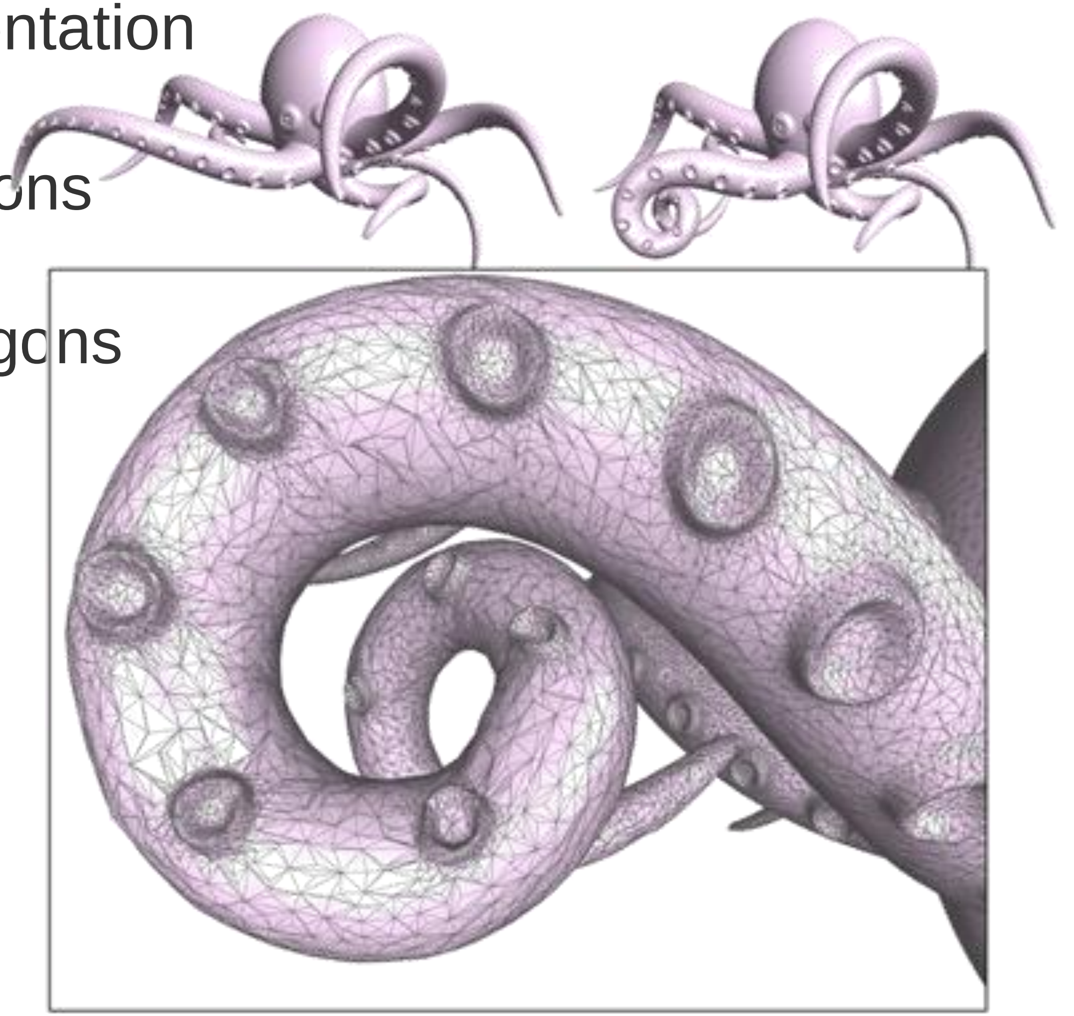
Geometric model representation

- A polygonal approximation typically consists of a discrete set of points of the continuous surface, which are connected via edges



  - Constructing the connectivity (edges) for a given set of points is called triangulation

Geometric model representation



- Polygonal approximations are typically adaptive, i.e., more/smaller polygons are used in regions of high curvature

Geometric model representation

- How do we store (represent) a triangle mesh
  - Geometry: where are the vertices located in 3D space
  - Topology: how are the vertices/polygons connected
  - Explicit representation:
    store three vertices for each of the *n* triangles
    *n**3*3 = 9*n* floats
  - High redundancy

t	v_i	v_j	v_k
0	1.0, 1.0, 1.0	-1.0, 1.0, -1.0	-1.0, -1.0, 1.0
1	1.0, 1.0, 1.0	-1.0, -1.0, 1.0	1.0, -1.0, -1.0
2	1.0, 1.0, 1.0	1.0, -1.0, -1.0	-1.0, 1.0, -1.0
3	1.0, -1.0, -1.0	-1.0, -1.0, 1.0	-1.0, 1.0, -1.0

Geometric model representation

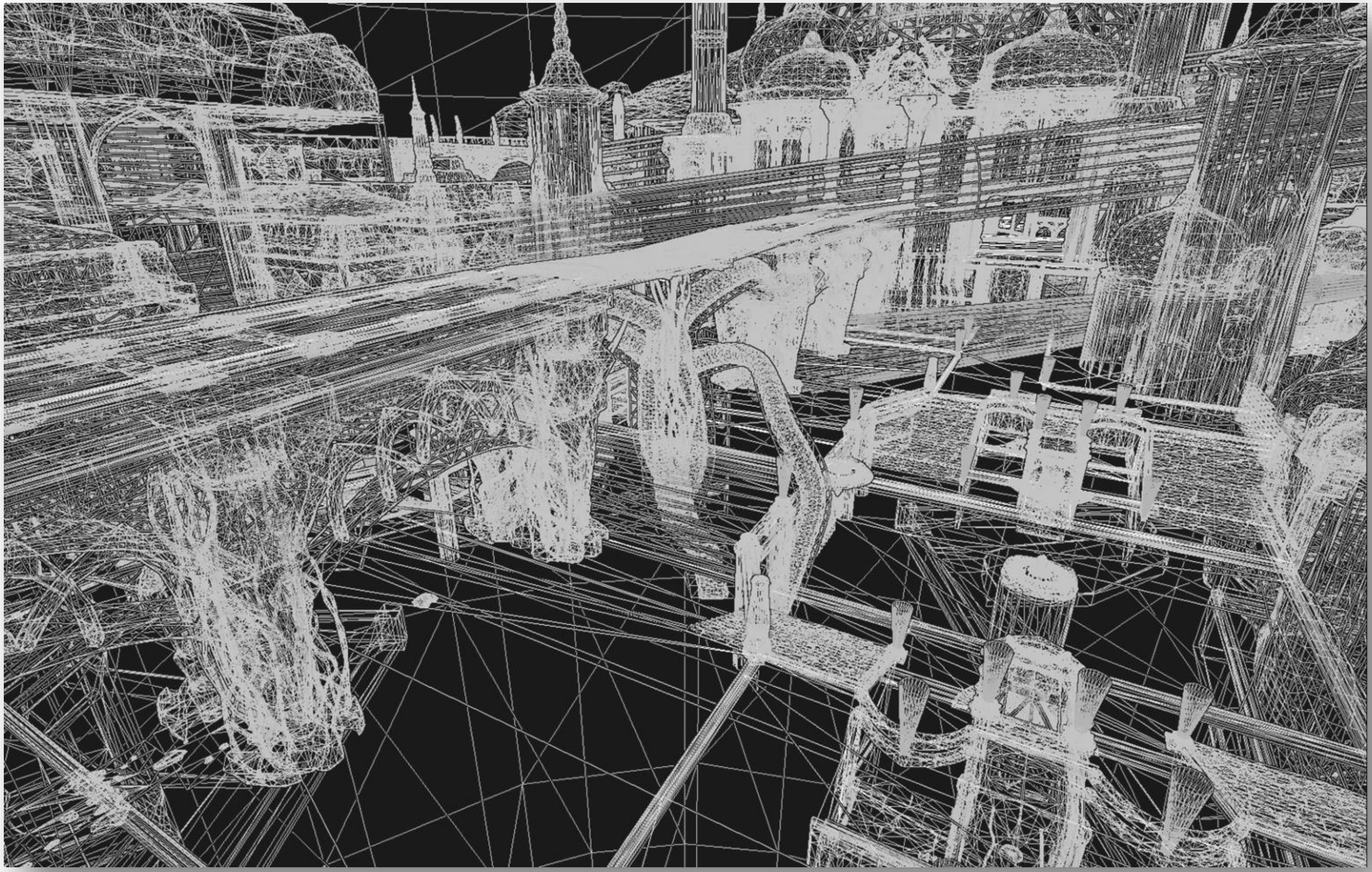
- Less redundant triangle mesh representation
  - Shared vertex („indexed face set“) representation
  - E.g., as in . **OBJ, OFF** files
  - Array of coordinates of all vertices (3*n* floats)
  - Array of triangles with indices into the vertex list (3*n* integers)

v	x	y	z
0	1.0	1.0	1.0
1	-1.0	1.0	-1.0
2	-1.0	-1.0	1.0
3	1.0	-1.0	-1.0

t	i	j	k
0	0	1	2
1	0	2	3
2	0	3	1
3	3	2	1

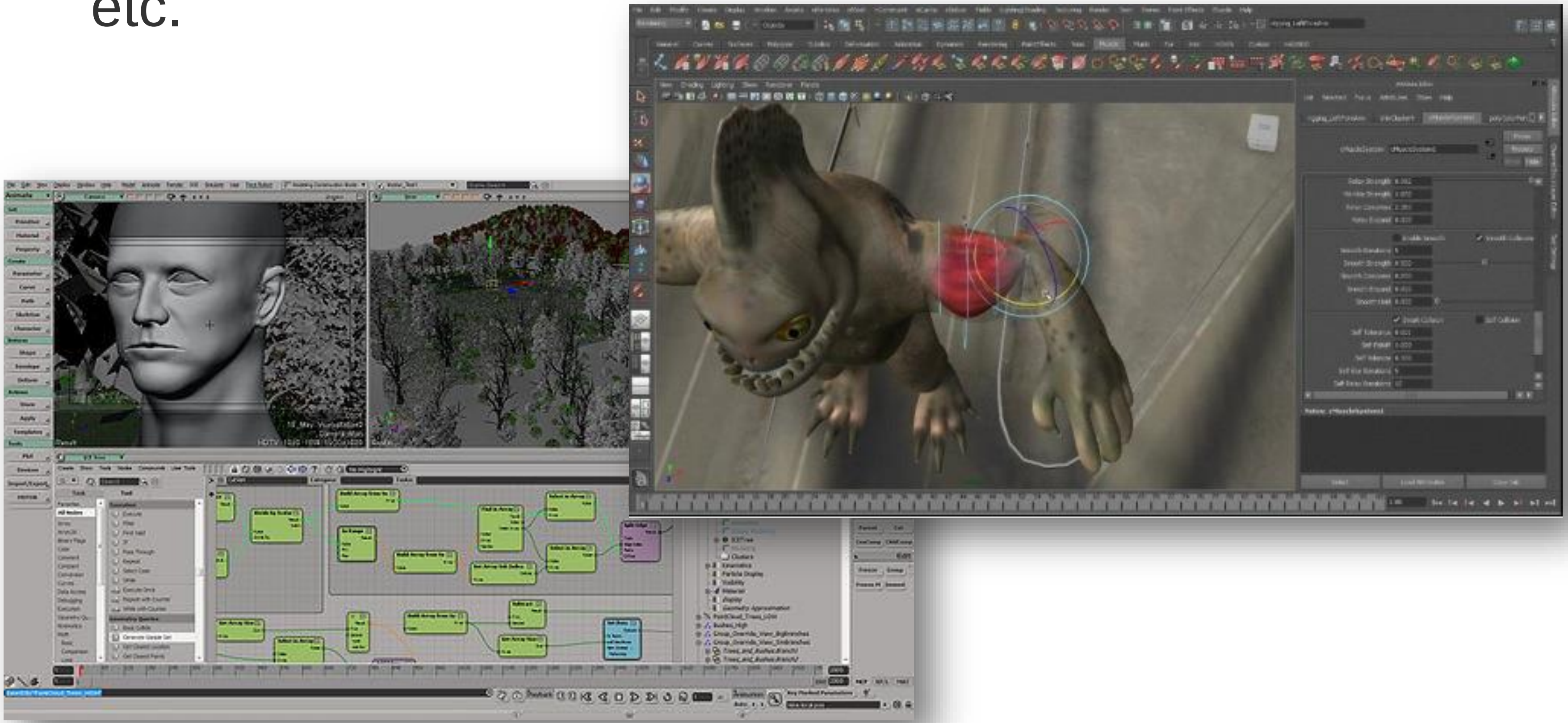
  - Use, eg. Meshlab, to view .obj files http://meshlab.sourceforge.net/

Geometric model representation – the wireframe



Geometric modelling

- Using modelling software like Blender, Maya, 3Ds-Max etc.



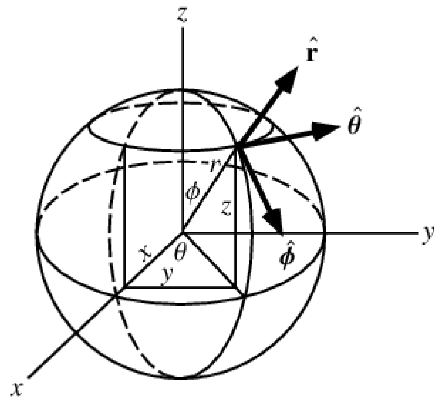
Procedural geometric modelling

- Wikipedia: Procedural modeling is an umbrella term for a number of techniques in computer graphics to – automatically – create 3D models and textures from sets of rules
- Example: a procedural unit sphere

$\theta: [0,2\pi]$ Azimuth

$\phi: [0,\pi]$ Zenith

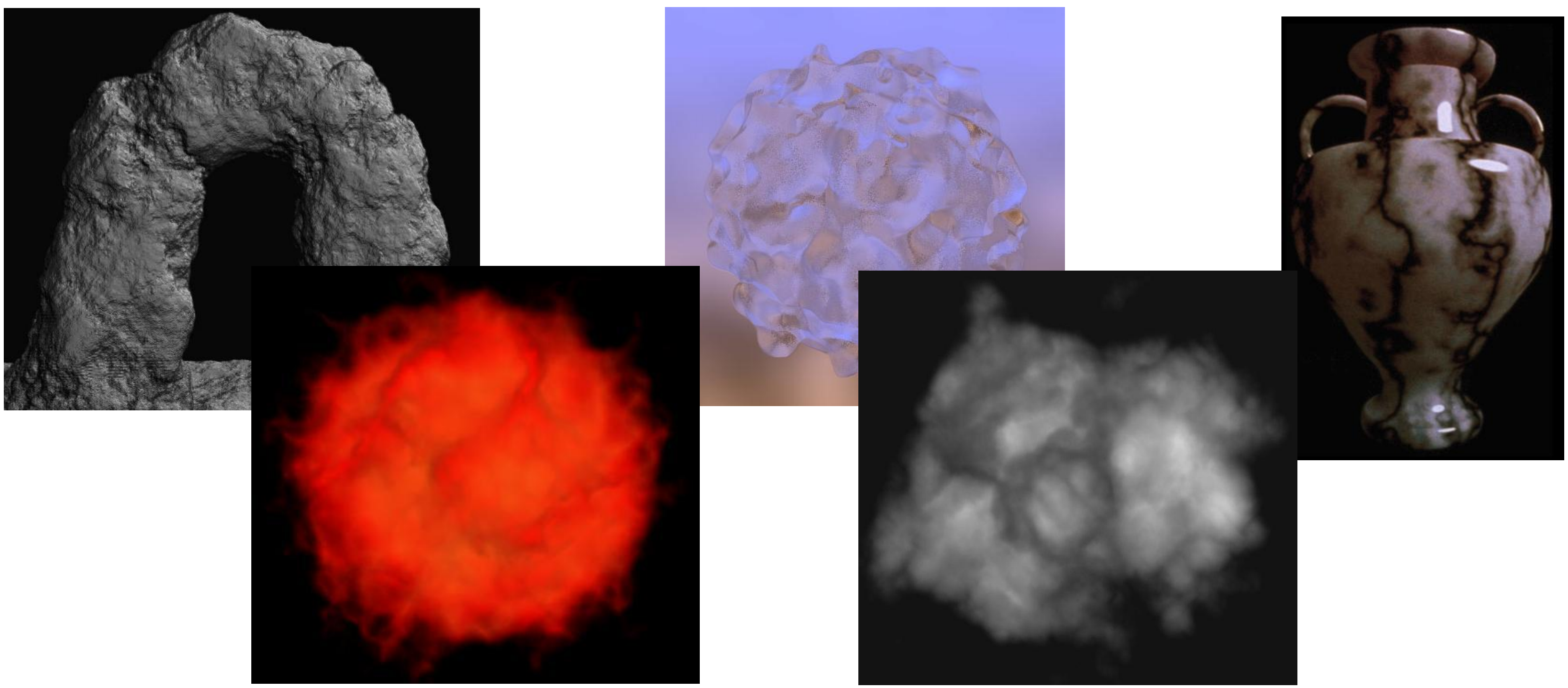
$$x(\theta,\phi) = \cos(\theta) \cdot \sin(\phi)$$
$$y(\theta,\phi) = \sin(\theta) \cdot \sin(\phi)$$
$$z(\theta,\phi) = \cos(\phi)$$



https://en.wikipedia.org/wiki/Procedural_modeling

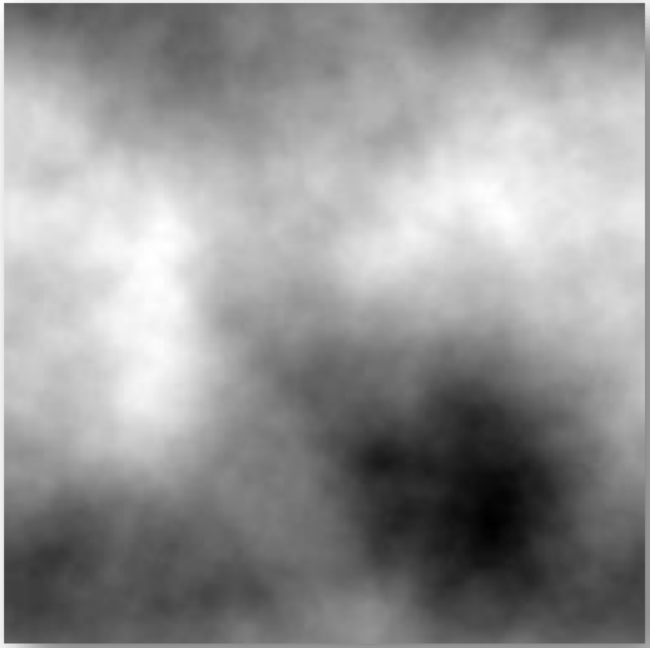
Procedural geometric modelling

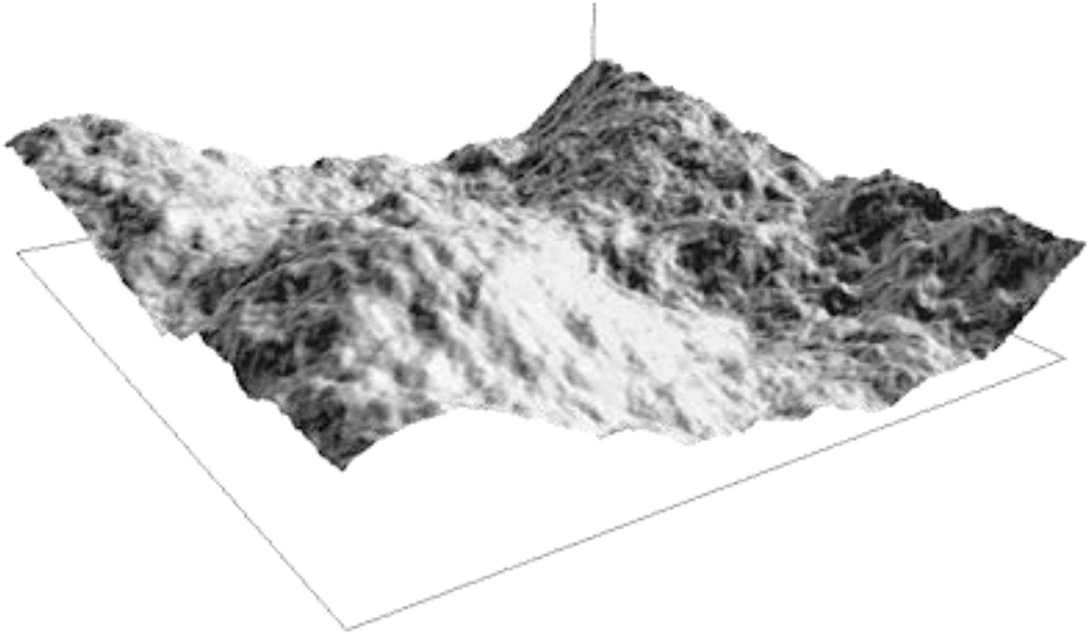
- Noise-based approaches to simulate structural variations as in reality



Procedural geometric modelling

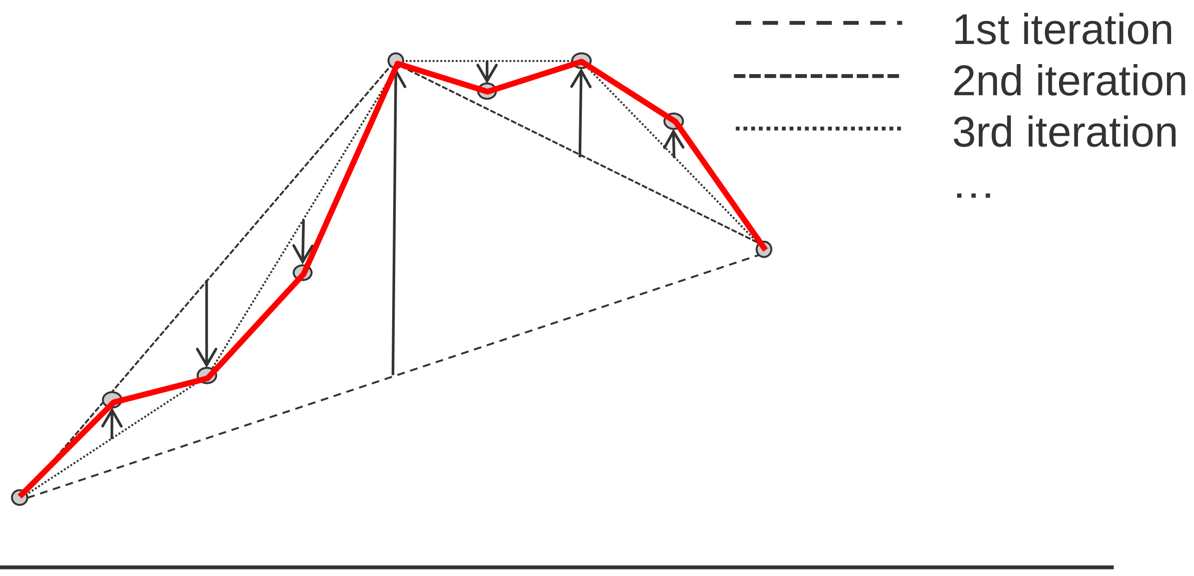
- Terrain modelling: generate values ∈ (0,1) over a 2D domain and draw as a height field





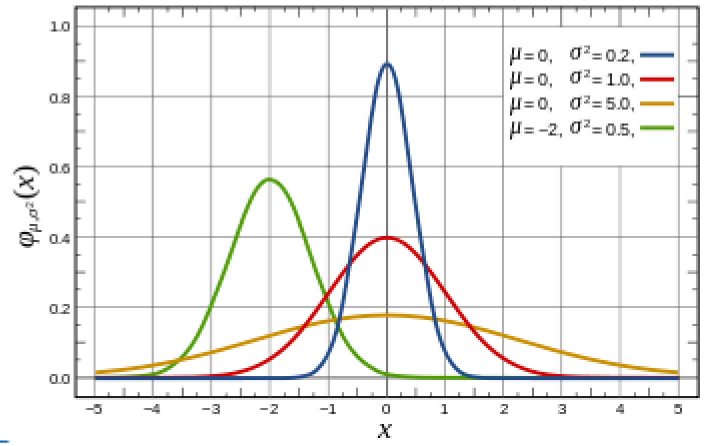
Procedural modelling

- Terrain modelling via random midpoint displacements
  - Interpolation and random displacements
  - Ever smaller displacements in each iteration



Procedural modelling

- Random displacements
  - Modelled via random numbers with a certain distribution; distribution defines the probabilities of the occurrence of values of a random variable
  - E.g., uniform [0,1] distribution: all values in [0,1] occur equally likely
  - E.g., normal distribution with expectation value μ and standard deviation σ
    - μ: the weighted average of all possible values
    - σ: the variation (spread) from the expectation value



Often used is the normal distribution $N(\mu = 0, \sigma^2)$

Procedural modelling

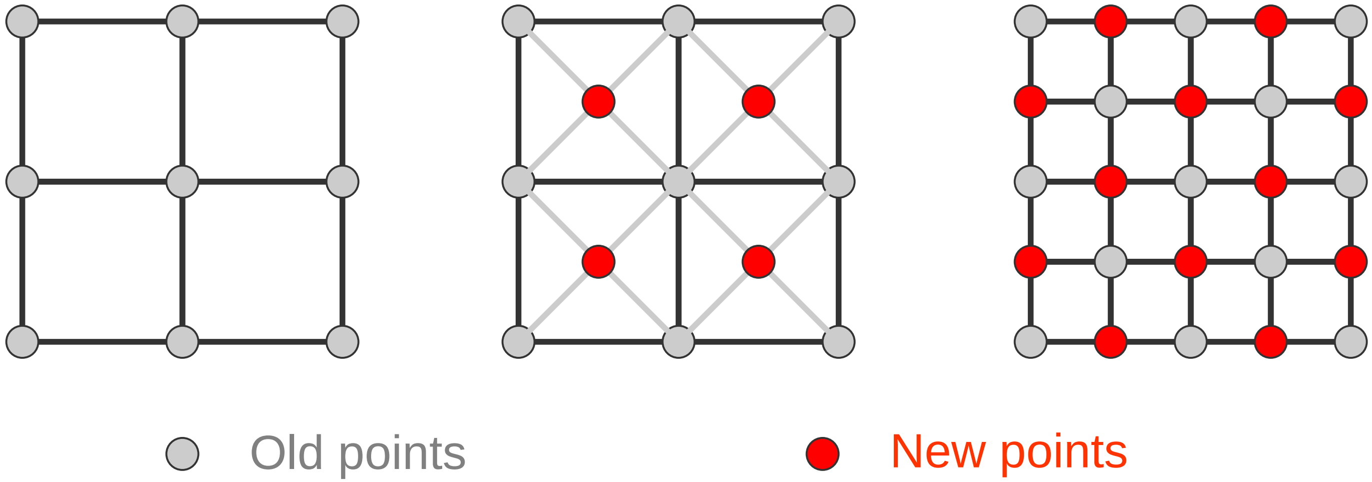
- Random displacements D_i (*i*: subdivision level)
  - Modelled via normal distributed random numbers with expectation value $\mu[D_i] = 0$ and standard deviation $\sigma[D_i] = \frac{\sigma_{i-1}}{2^H}$
  - Starting with σ_0 at the first level, ever decreasing random displacements are generated at increasing subdivision levels; *H* controls fall-off and thus roughness of terrain
  - Computation from $N(0,1)$ distributed random numbers via $N(0,\sigma^2) = \sigma N(0,1)$
  - In C++:

```
#include <random>

template< class RealType = double >class normal_distribution;
```

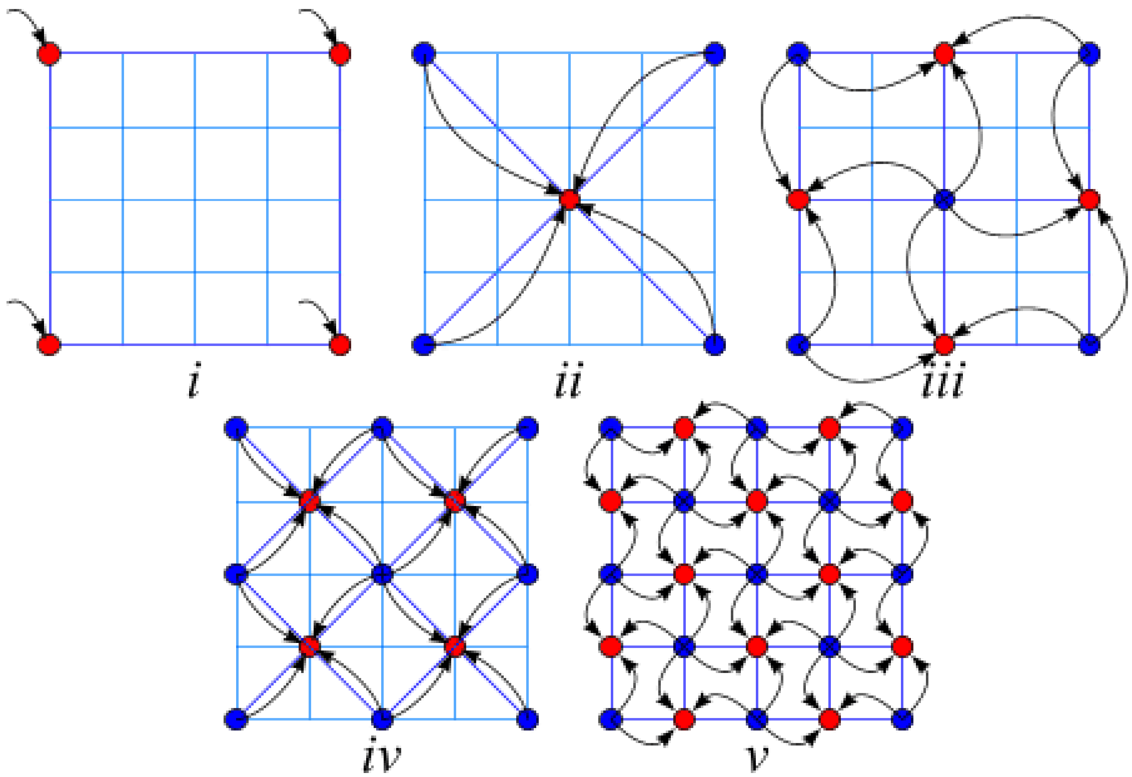
Procedural modelling

- Terrain generation
  - Interpolate new values from adjacent values
  - Add random displacement



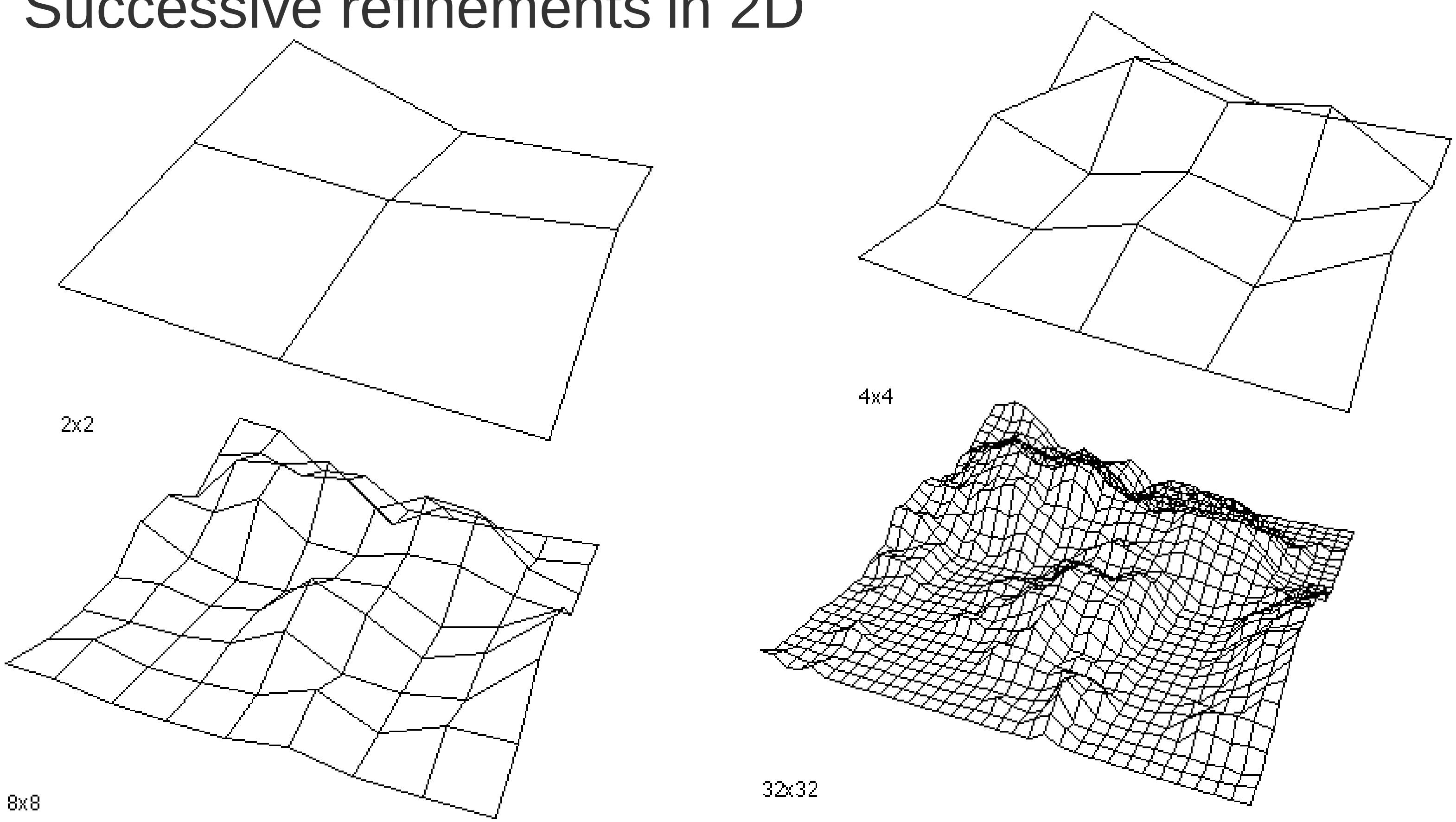
Procedural modelling

- Terrain generation – the Diamond Square algorithm



Procedural modelling

- Successive refinements in 2D



Procedural modelling

- From the 2D height raster to triangles; replace each quadrilateral by 2 triangles: