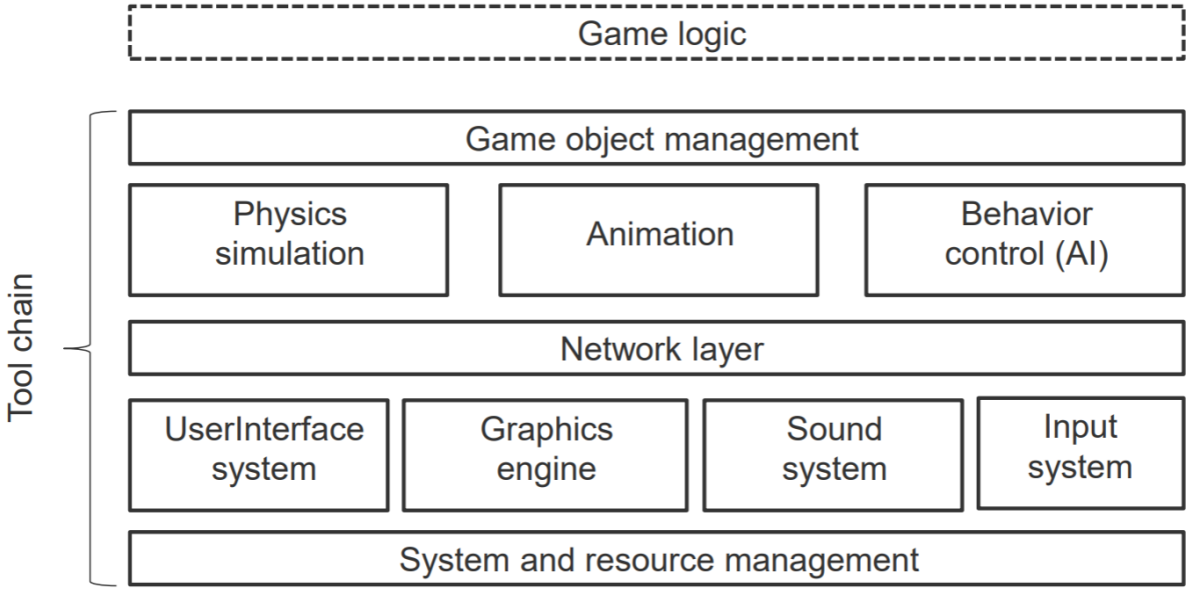
Robin Weiland

EIST Summary

01 Game Engine Design

Game logic	Data	Game Engine
algorithms realizing the game rules	various data used by the engine	other algorithms, "independent" of the game logic and data

Game engine components



Game Engine Data

- Media assets geometric models, textures, animations, sounds
- Level data objects in a level and how they interact
- Object configuration data physical properties and behavior control data
- User Interface configuration data types of input devices and their mapping
- Engine configuration data drivers, resolution, ...

02 Geometry

Geometric objects

- characters, items, props, game levels, ...
- Usually created via **geometric modelling tools** (blender, maya, ...)
- Modelling:
  - **shape modelling:** shape representation by placing *geometric primitives*
  - **appearance modelling:** material properties like color, reflection properties, textures, etc.

Geometric model representation

> *Models are polygonal approximations of continuous objects.*

Polygons

- **points**, typically in 3D-space ($\mathbb{R}^3$), represented by $x/y/y$-coordinates, called **vertex**
- **valence**: number of adjacent vertices of a given vertex
- **edges**, line segments connecting two **vertices**
- **polygon**, interior of a closed planar connected series of line segments

  (edges do not cross each other and each vertex is connected with exactly two edges)

Polygon mesh

- Surface composed of polygons (**faces**)
- **Hollow**, if mesh is not closed

Polygonal approximation

- Approximation of a continuous surface by a polygon mesh.
- Typically consists of a **discrete set of points** of the continuous connected via edges.
- Constructing the connectivity (edges) for a given set of points is called **triangulation**.
- Usually **adaptive**: more/smaller polygons $\longrightarrow$ regions with high **curvature**

Mesh storage / representation

- **Geometry** vertices' location in 3D-space
- **Topology** connection of vertices & polygons

Explicit representation

Store three vertices for each of the F triangles: $F * 3 * 3 = 9F \quad floats$ (**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

Triangle Mesh representation

- **Shared vertex** ("indexed face set") representation
- Used in e.g. *.obj*, *.off* files
- Array of all vertices (3V floats)
- Array of triangles with indices into the vertex list (3F 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

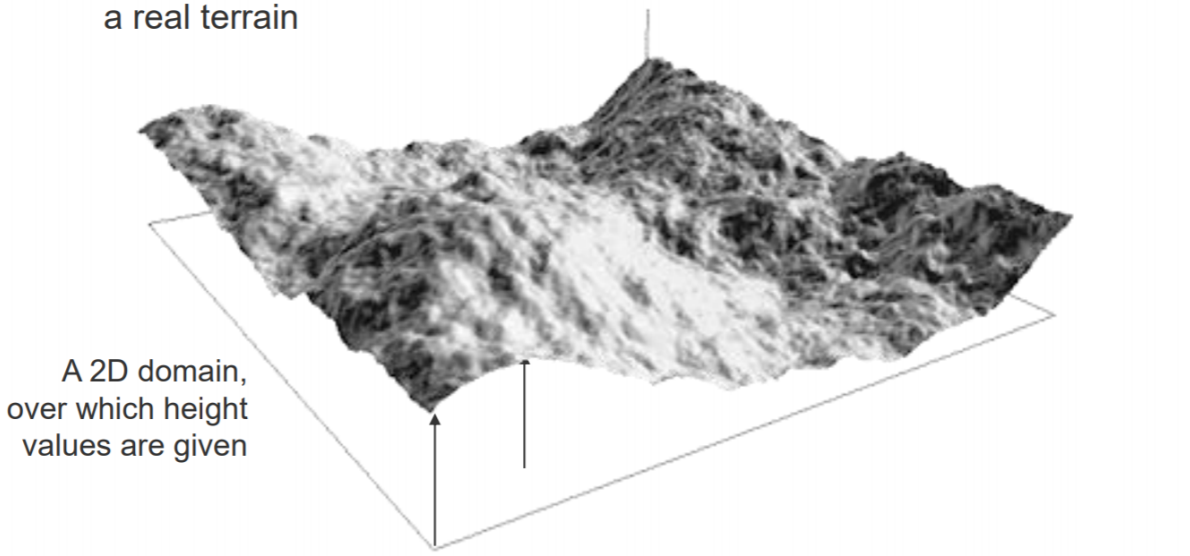
Procedural Modelling

> *Number of techniques to **automatically** create 3D models and textures from sets of rules*

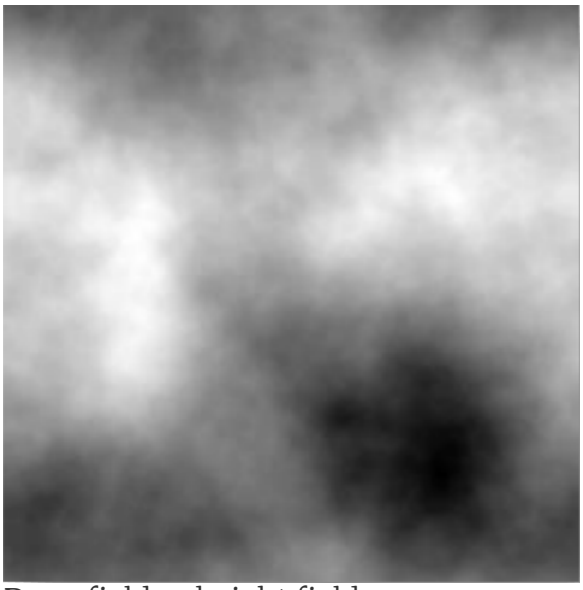
Example: sphere generation after trigonometric rules.

Procedural terrain modelling

Create a piece of landscape, i.e., a **height field**, that has the shape of a real terrain



- Generate a random 2D "density field" and interpret values as height values over a 2D domain



- Draw field as height field

TODO diamond square algorithm