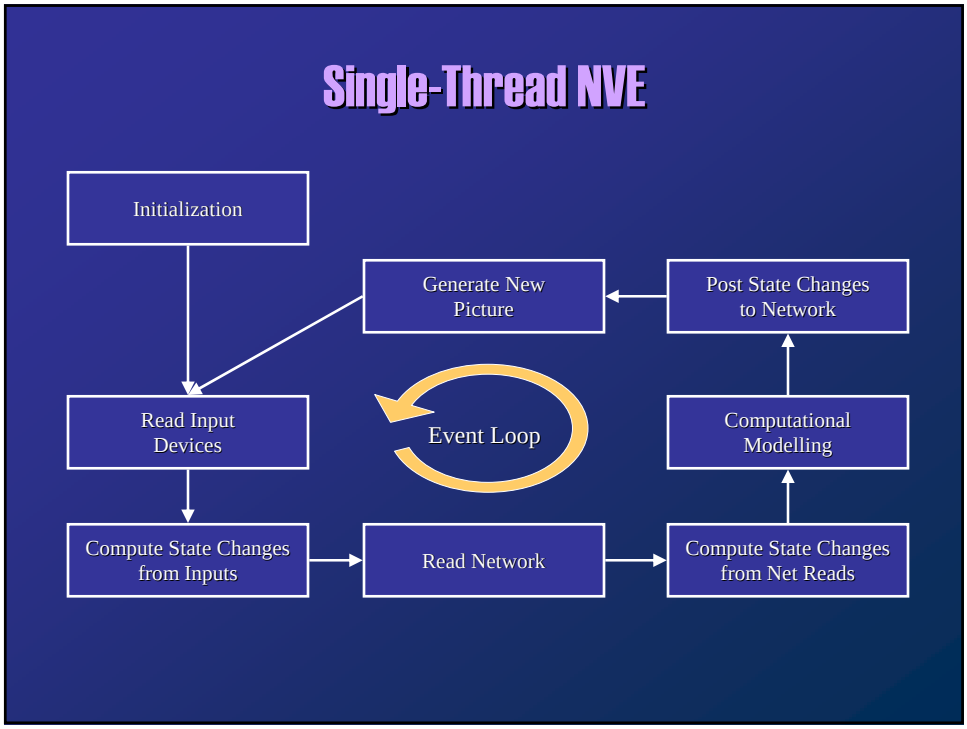
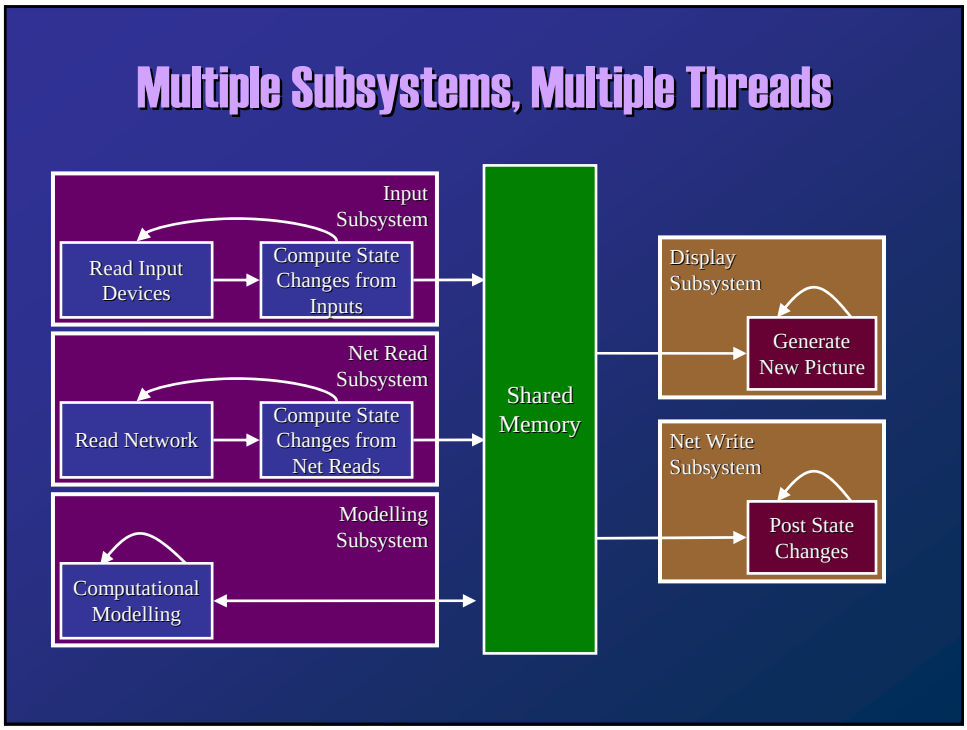

Single-Thread NVE
Initialization
Generate New Picture
Post State Changes to Network
Read Input Devices
Event Loop
Computational Modelling
Compute State Changes from Inputs
Read Network
Compute State Changes from Net Reads

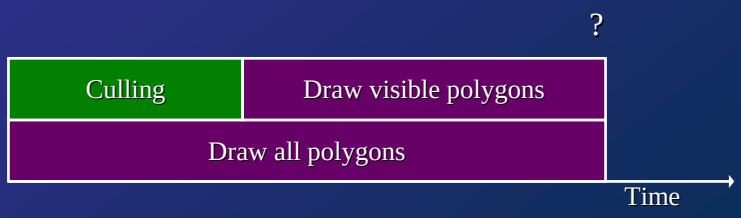

Multiple Subsystems, Multiple Threads
Input Subsystem
Read Input Devices
Compute State Changes from Inputs
Net Read Subsystem
Read Network
Compute State Changes from Net Reads
Modelling Subsystem
Computational Modelling
Shared Memory
Display Subsystem
Generate New Picture
Net Write Subsystem
Post State Changes

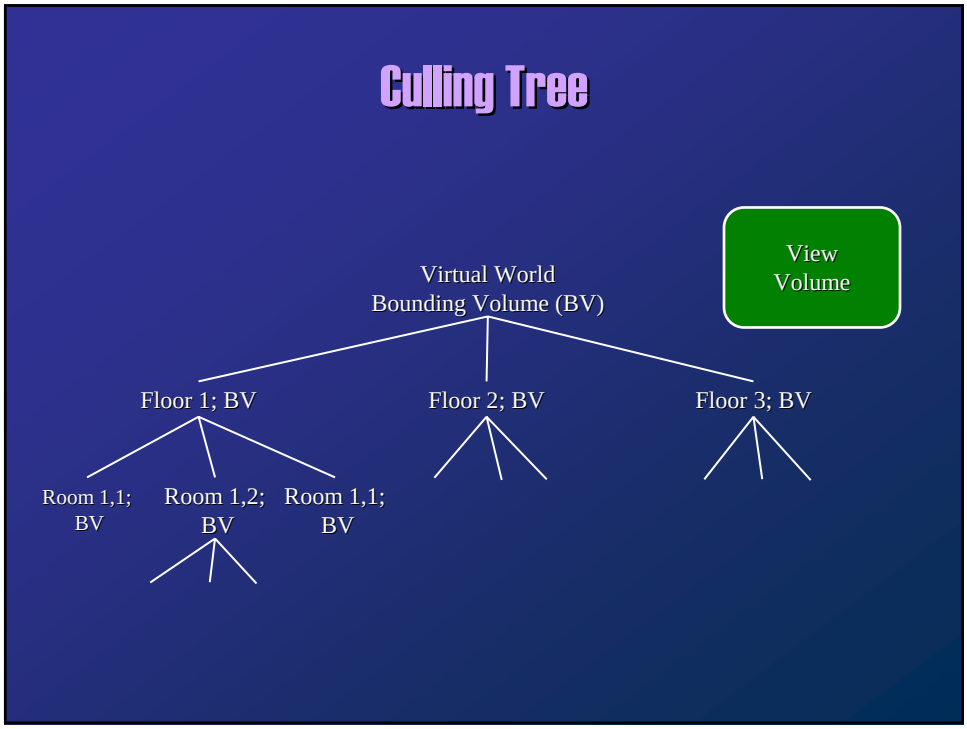
Real-Time Rendering

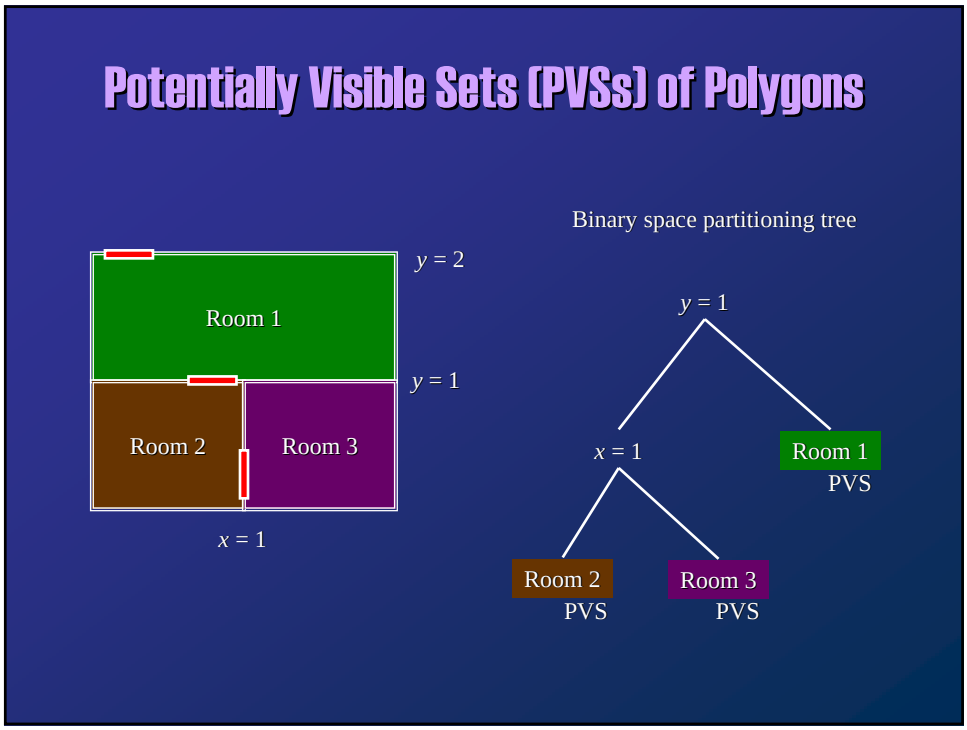
- Key problem: limitations in the performance of graphics hardware
  - frames per second
  - polygons per second
- Polygon culling
- Level-of-detail processing

Polygon Culling

- Reduce the number of processed polygons
  - determine which polygons do not need to be drawn
- A wealth of research and methods exists
- Assumes that the underlying 3D model remains quite static
  - changes in the model ⇒ changes in the culling data structures

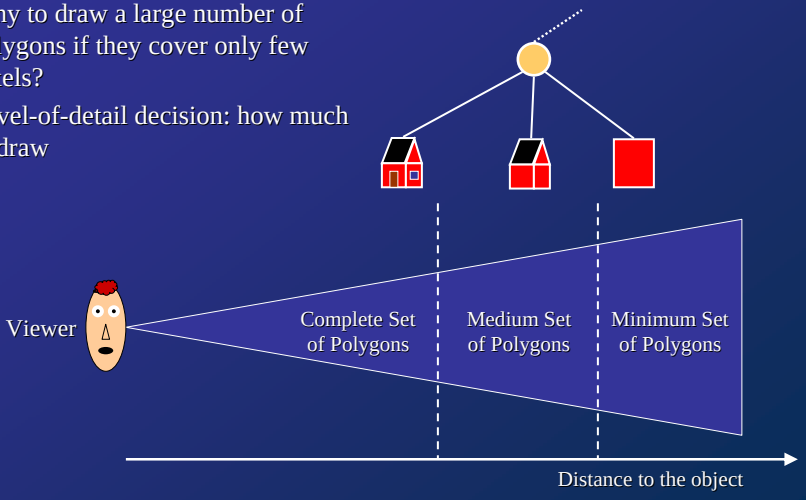



Culling Tree
View Volume
Virtual World Bounding Volume (BV)
Floor 1; BV
Floor 2; BV
Floor 3; BV
Room 1,1; BV
Room 1,2; BV
Room 1,1; BV


Potentially Visible Sets (PVSs) of Polygons
Binary space partitioning tree
Room 1
Room 2
Room 3
y = 2
y = 1
x = 1
y = 1
x = 1
Room 1 PVS
Room 2 PVS
Room 3 PVS

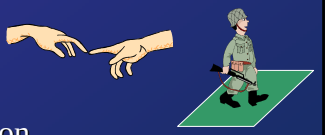
Levels of Detail (LOD)

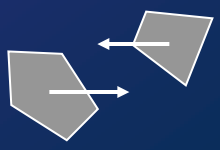
- Why to draw a large number of polygons if they cover only few pixels?
- Level-of-detail decision: how much to draw



Real-Time Collision Detection and Response

- Interacting with the VE
  - touching, grasping, standing,…
- Take some action in response to the collision
- Is there an intersection with the polygons of an object and the polygons of any other object?
  - test bounding boxes
  - utilize hierarchical data structures
- Where are the precise contact points?





Real-Time Collision Detection Solutions

- Approaches to collision detection
  - geometric reasoning
  - bounding volume hierarchy
  - analytical methods
  - hybrid
- Fast, approximate collision detection
  - ownship: static object collisions
  - ownship: moving object collisions
- Fast, accurate collision detection

Fast, Approximate Collision Detection

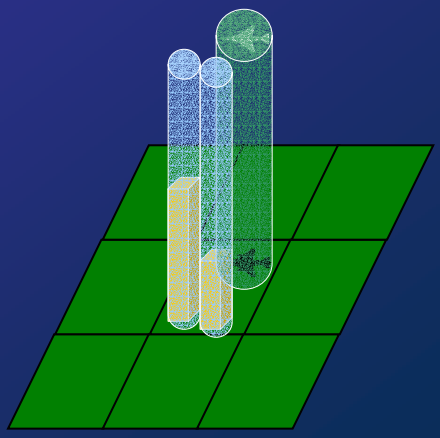
- Important to recognize that a collision has occurred
- The precise location of the collision is unimportant
- Example: NPSNET
  - moving objects can collide with each other and with fixed, static objects
  - upon collision over a certain speed ⇒ the moving object dies
  - no sophisticated physics
- Ownship = the local player in the VE
  1. Moving object (ownship) against static objects
  2. Moving object (ownship) against moving objects (other players)
- Up to the ownship to report its collisions and its death

Ownship: Static Object Collisions

- Occurs when an object has moved and its position is updated
- Reduce the set of static objects that must be considered

1. Is the ownship below the a threshold elevation?
2. Calculate 2D distance to all objects in the grid square
3. Is the ownship's ground elevation less than the height of the static object?

- Issue a detonation PDU or an entity state PDU

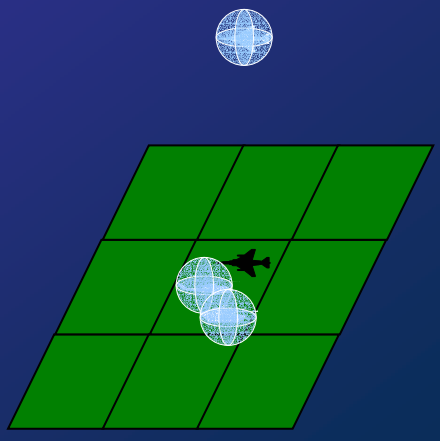


Ownship: Moving Object Collisions

- The ownship did not collide with any static object
- Reduce the collision comparison space

1. Check current and neighbouring squares
2. Calculate 3D distances to objects
3. Cast a ray from the ownship's origin to the moving object's origin

- The host managing moving object also performs collision detection and issues PDUs



Fast, Accurate Collision Detection

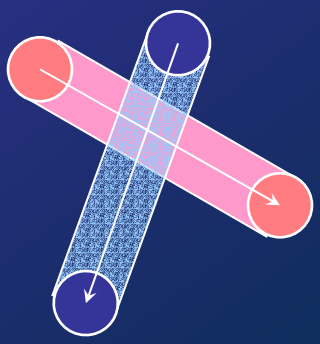
- Sweep-and-prune algorithm
- An axially aligned 3D bounding box for each object
- Sort the bounding boxes
- Are the bounding boxes overlapping?
  - for 3D bounding boxes to collide, their projections must overlap
- Are the the convex hulls overlapping?
- Compute the actual area of collision

Problems of Collision Detection in NVEs

- Who determines collision in an NVE?
- The object that has collided
  - DIS does not require that the hosts use the same collision detection algorithm
  - what if one decides to die, whilst another decides that there was no collision
  - fair play requires a standard for collision detection
- What about collisions that happen
  - in between time steps, or
  - for dead-reckoned objects?

Collisions in between Time Steps

- The objects are moving too fast
- The time steps between frames are too large
- Requires additional computation



Collisions for Dead-Reckoned Objects

- The ownship may determine collision with a dead-reckoned object and issue a packet
- The object collided with is at a slightly different actual position
  - no collision
  - collision with different results

- Mechanism for establishing an agreement on which the objects reach an acceptable conclusion
- Recognize arriving packets that indicate mutual collision
  - the object that missed the collision must also realize it
  - problem between the time of real collision and the learning time
  - how to correct the past?

Computational Resource Management

- How to allocate processor time for the processes
  - do we leave it to the operating system?
- A blocked thread should yield the processor to the threads in waiting
- Subsystems in separate threads
  - input subsystem
  - net read subsystem
  - display subsystem
  - net write subsystem
  - modelling subsystem