Health Issues
Accelerated Rendering

Lecture 4

RVAU - Realidade Virtual e Aumentada - EIC0070

2019/2020 - 1S

Filipe A. Rodrigues

Jorge A. Silva

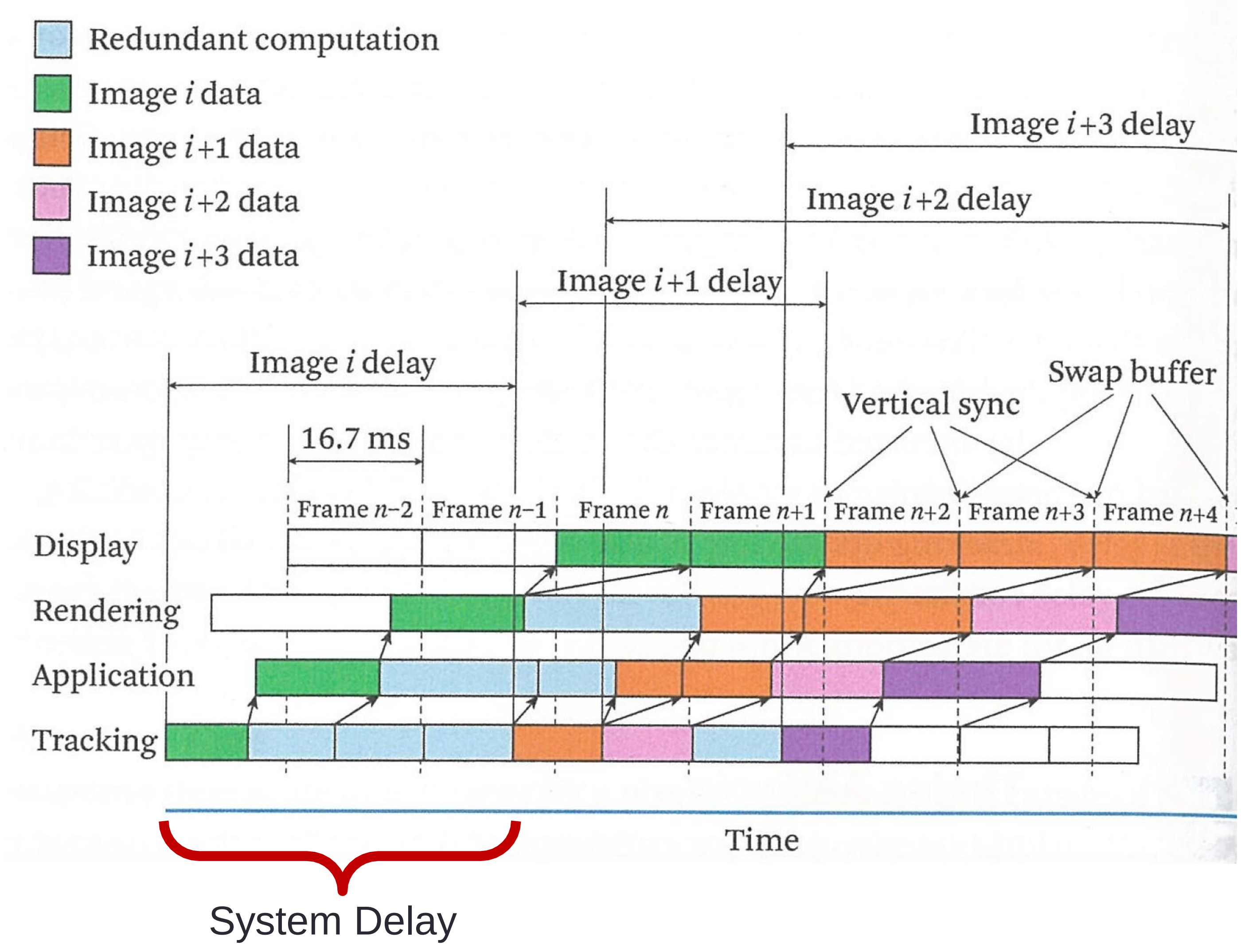
(adaptado de slides Rui Nóbrega, A. Augusto Sousa)

Overview

- Health and Latency Issues
- Accelerating Rendering

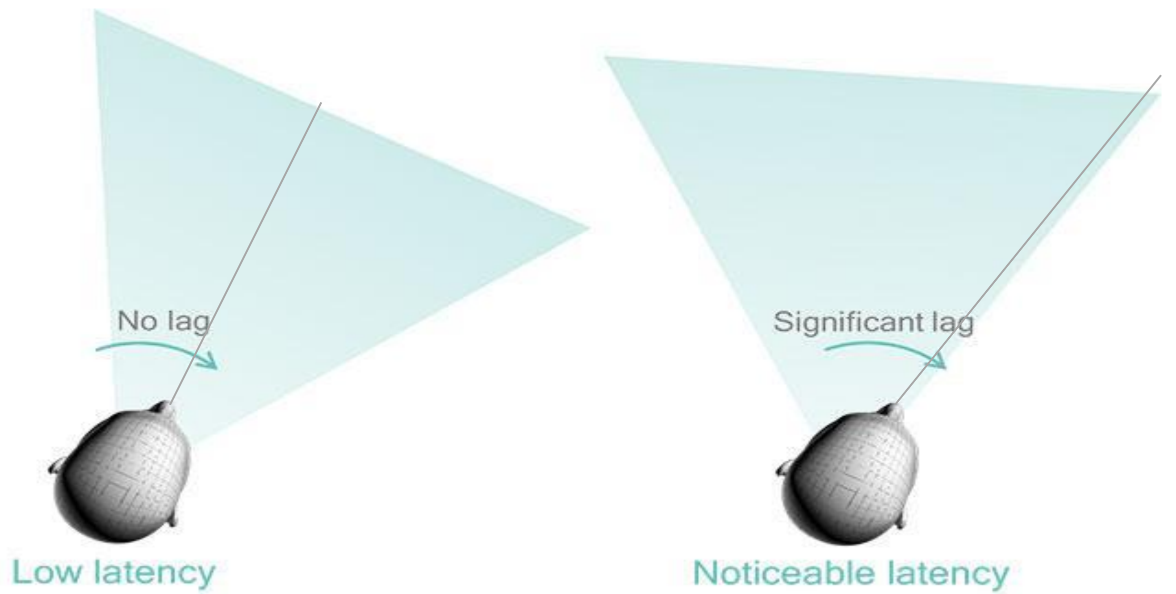
Health Issues

Back to Latency....



Latency

Latency is the **time** a **system takes** to **respond** to a user's **action**, the **true time** from the **start** of **movement** to the time a **pixel** resulting **from** that **movement responds** [Jerald, 2009].



Jerald, J. (2009). Scene-motion and latency-perception thresholds for head-mounted displays.

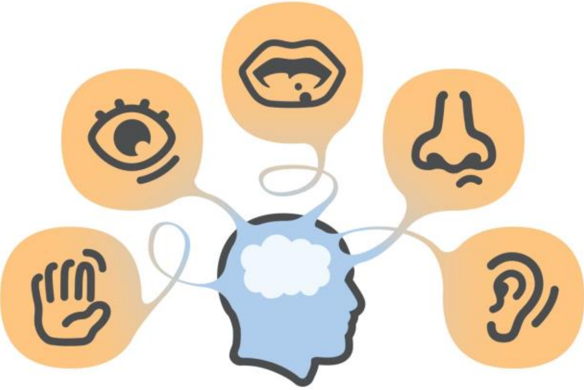
Latency

For low latencies

- **Below ~100ms**
- **Users** do **not perceive the latency directly**, **but** rather the **consequences** of **latency**...

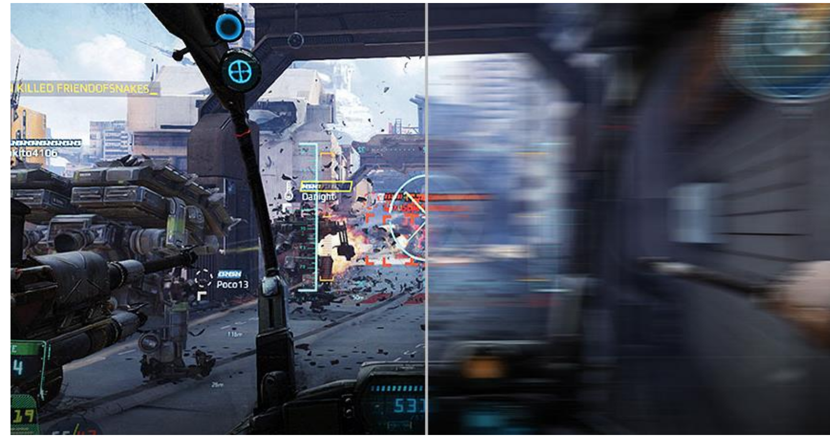
Latency in an HMD system causes **visual cues** to **lag** behind **other perceptual cues**, creating sensory conflict (motion sickness)

- e.g., visual cues get out of phase with vestibular cues (balance and spatial orientation)



Negative Effects of Latency

- Degraded Visual Acuity
  - **Scene** still **moving** when **head stops** = blur



- Degraded Performance
  - So and Griffin (1995) studied the **relationship between latency** and operator **learning** in an **HMD**
  - If **latency > 120 ms**, **training** does **not improve performance**
    - Subjects were unable to learn to compensate for these latencies in the task

So, R. H. Y., and Griffin, M. J. (1995). Effects of Lags on Human Operator Transfer Functions with Head-Coupled Systems. Aviation, Space and Environmental Medicine, 66(6), 550–556

Negative Effects of Latency

- Breaks-in-Presence
  - The **incorrect scene motion** can **distract** the **user**, **breaking** the **illusion**

- Negative Training Effects
  - Some studies shown that **latency** can have an unintended **decrease** in **performance** when **training** for a task
    - With desktop displays (Cunningham et al., 2001b)
    - And driving simulators (Cunningham et al., 2001a)

- Cunningham, D. W., Billock, V. A., and Tsou, B. H. (2001a). Sensorimotor Adaptation to Violations in Temporal Contiguity. PsychologicalScience, 12(6), 532–535. 145, 184
- Cunningham, D. W., Chatziastros, A., Von Der Heyde, M., and Bulthoff, H. H. (2001b). Driving in the Future: Temporal Visuomotor Adaptation and Generalization. Journal ofVision, 1(2), 88–98. 184

Negative Effects of Latency

Latency is the greatest cause of simulator sickness

Latency: Simulator Sickness

Central processing

- **Brain** gather data from **senses** to **form** a better **estimate** of the **world**…
- Uses **additional input** from an internal **mental model** (memories, expectations, etc…)

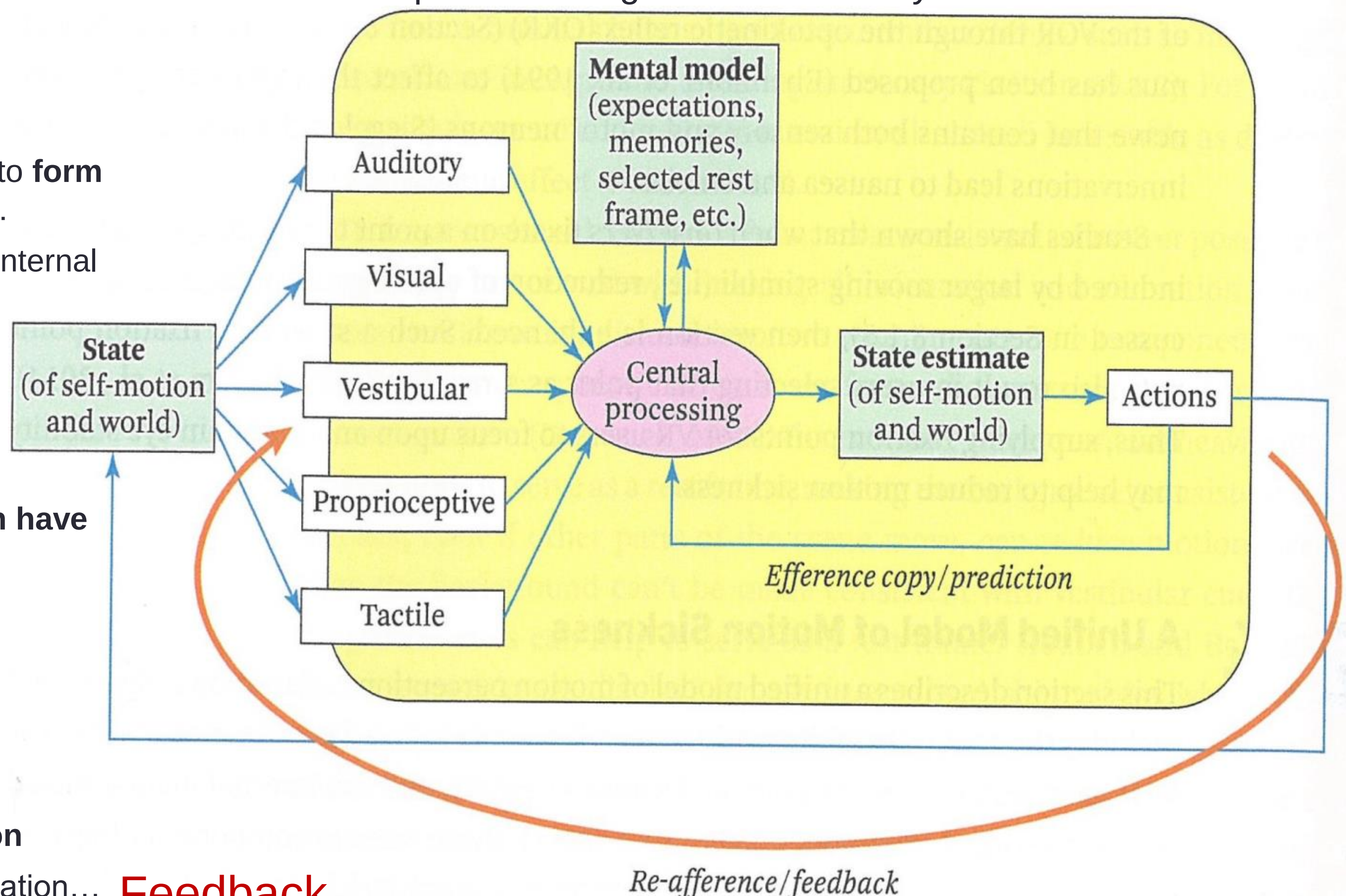
State estimation

- At **any given moment** the **brain have** an **estimate** of the world
- It **includes**:
    - what is **happening**
    - what **will happen**

Actions

- **Body** tries to **fit state estimation**
    - Change posture, eyes rotation…
- **Physiological** response: sweating, vomiting…

Visual input conflicting with vestibular system



Feedback

- **Perception of a visually stable world relies on prediction of how incoming sensory information will change due to one's action.**

Many Causes of Simulator Sickness

25-40% of VR users get Simulator Sickness, due to:

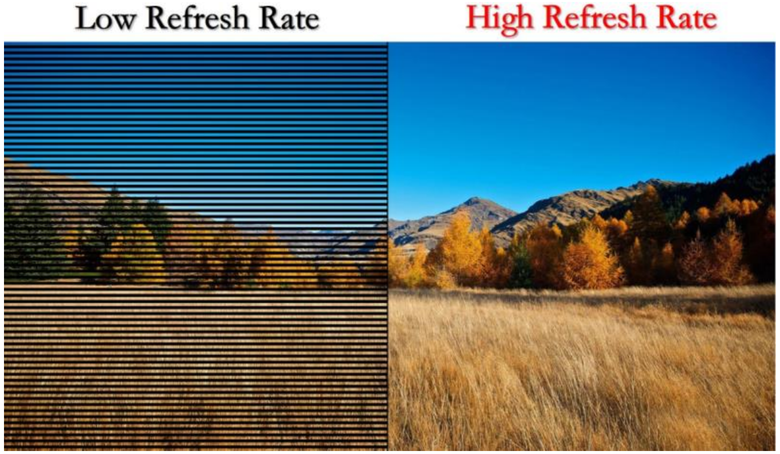
1. Latency
    - Major cause of simulator sickness
2. Field of View
    - A wider field of view can create more **periphery vection**
        - Vection is the **sensation** of **movement** of the body in space **produced** by **visual stimulation**

Many Causes of Simulator Sickness

25-40% of VR users get Simulator Sickness, due to:

3. Tracking accuracy/precision
    - **Low accuracy** can **drift over time,** accumulating error…
    - Leading the user to **see the world** from an **incorrect viewpoint**

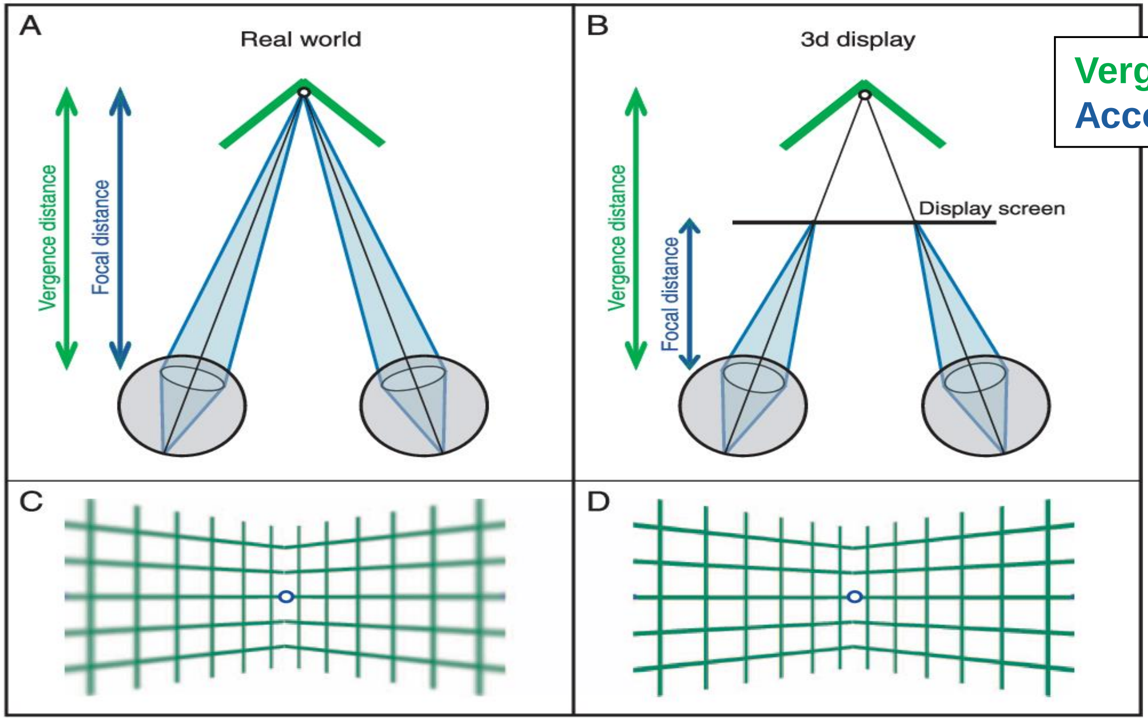
4. Refresh Rate/Flicker
    - Flicker/low refresh rate is distracting, creates eye fatigue, and can cause seizures



Many Causes of Simulator Sickness

25-40% of VR users get Simulator Sickness, due to:

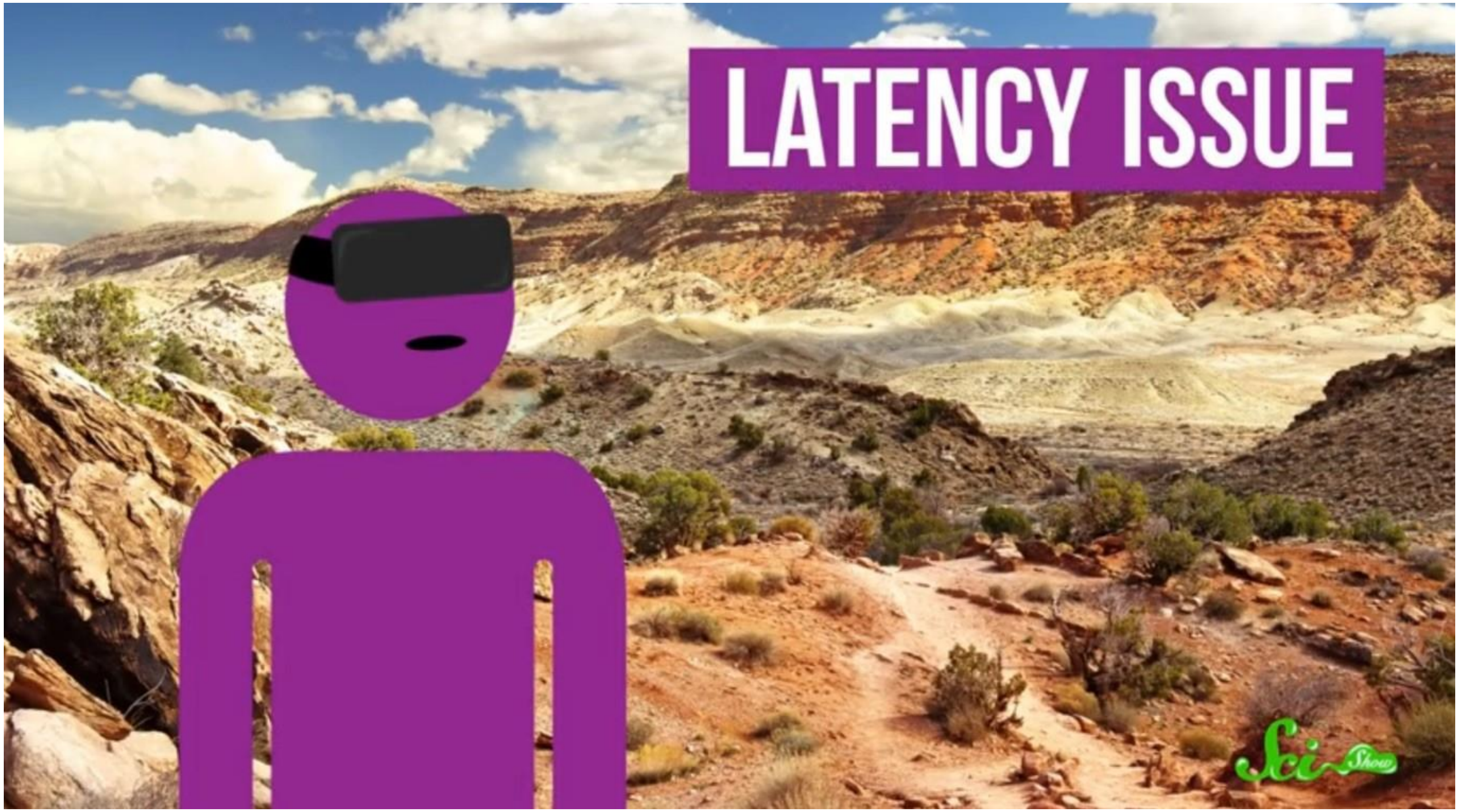
5. Vergence-Accommodation Conflict
   - Creates eye strain over time



Vergence
Accomodation

6. Eye separation
   - If IPD not matching to inter-image distance then discomfort

Solve Motion Sickness?



- https://www.youtube.com/watch?v=BznbIlW8iqE

Cyber Sickness Questionaires

- How to evaluate cyber sickness?
  - VR sickness is difficult to measure
    - Multiple symptoms
    - It varies accross individuals...
    - Weak effects
    - Individuals tend to adapt
- Solution: Using questionnaires
  - It is a standard practice to **only give** a simulator sickness **questionnaire after** the **experience** to remove bias
- Example:
  - **The Kennedy Simulator Sickness Questionnaire (SSQ)**

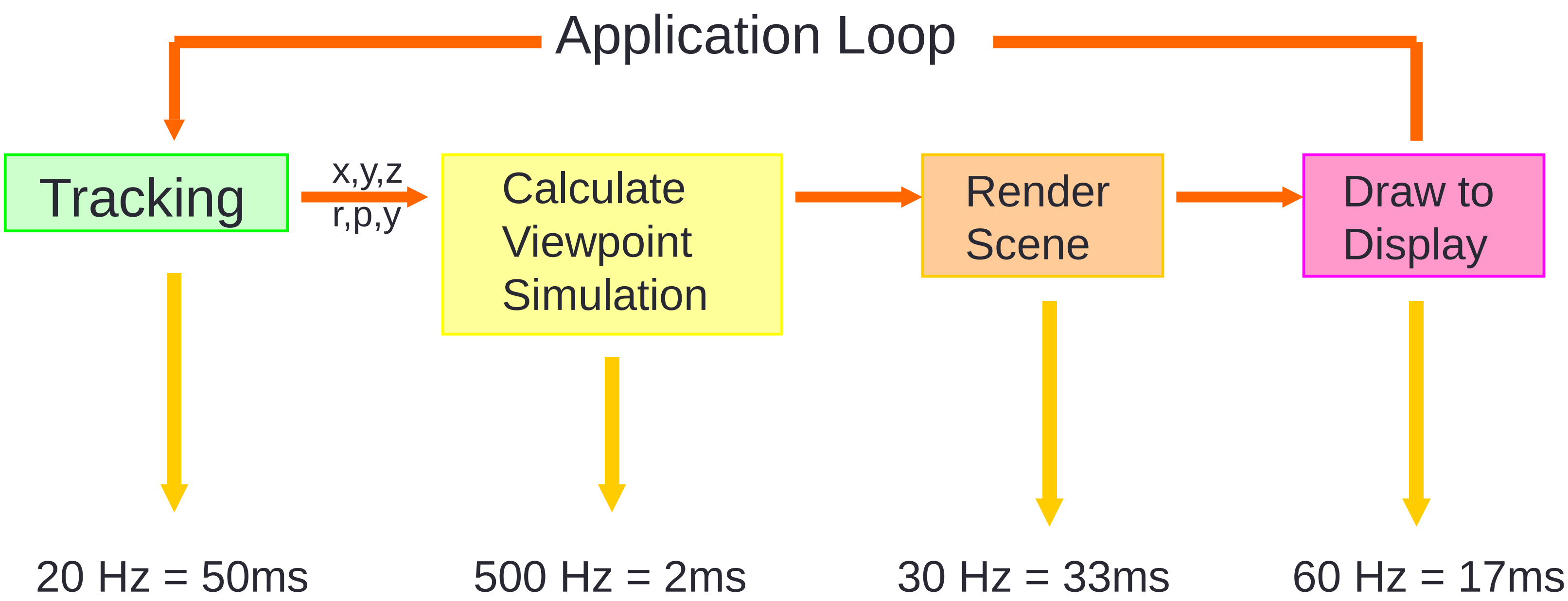
- Kennedy, R. S., and Fowlkes, J. E. (1992). Simulator Sickness Is Polygenic and Polysymptomatic: Implications for Research. The International Journal of Aviation Psychology. DOI: 10.1207/s15327108ijap0201_2.
- Young, S. D., Adelstein, B. D., and Ellis, S. R. (2007). Demand Characteristics in Assessing Motion Sickness in a Virtual Environment: Or Does Taking a Motion Sickness Questionnaire Make You Sick? In IEEE Transactions on Visualization and Computer Graphics (Vol. 13)

How to Reduce System Delays

- Use faster components
  - Faster CPU, display, etc...
- Reduce the apparent lag
  - Take **tracking measurement just before rendering**
  - Remove tracker from the loop
- Use predictive tracking
  - Use fast **inertial sensors** to **predict** where user will be looking
  - Difficult due to erratic head movements… **can create** some **catastrophic errors**

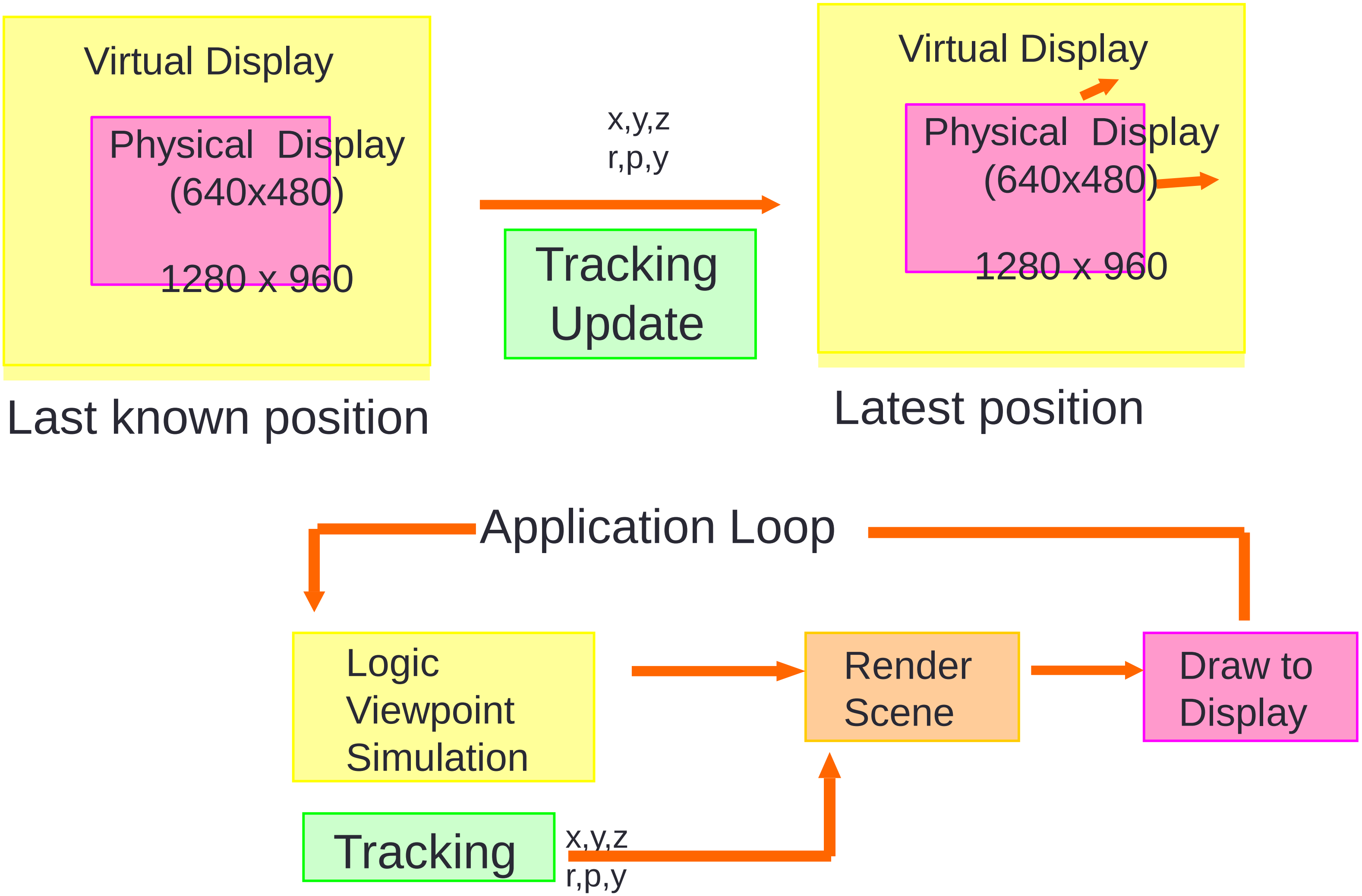
Jerald, J. (2004). *Latency compensation for head-mounted virtual reality*. UNC Computer Science Technical Report.

Reducing Apparent Lag: Typical System Delays



- Total Delay = 50 + 2 + 33 + 17 = 102 ms
  - **1 ms delay** = **1/3 mm error** for **object drawn** at **arms length**
  - So we have a **total of 33mm error** from when **user begins moving to when object drawn**

Reducing Apparent Lag



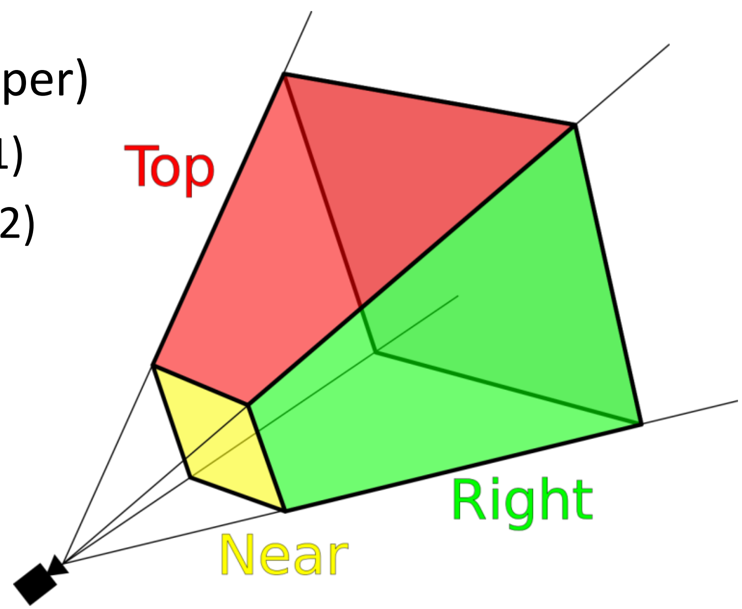
Jerald, J. (2004). *Latency compensation for head-mounted virtual reality*. UNC Computer Science Technical Report.

Accelerating Rendering

Based on:
Pedro Pires, Dynamic Algorithm Binding for Virtual Walkthrough, 2001

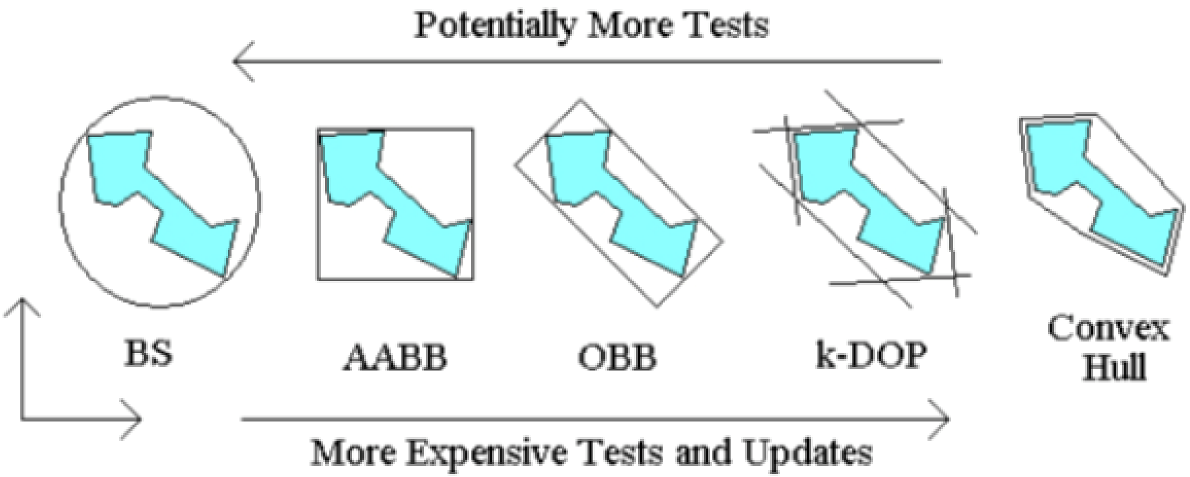
View-Frustum Culling

- View-frustum culling techniques aim to **quickly eliminate objects outside** the **current view-volume**
  - Z-transform all geometry and then clip any polygons whose vertices fall outside of the viewport
  - or whose Z value is greater or less than the near/far clipping planes
- They perform **gross visibility tests on sets of objects** prior to sending them to the graphics pipeline
- How? (see chapter 2 annexed paper)
  - Bounding Volume Hierarchies (2.1)
  - Spatial Partitioning Hierarchies (2.2)



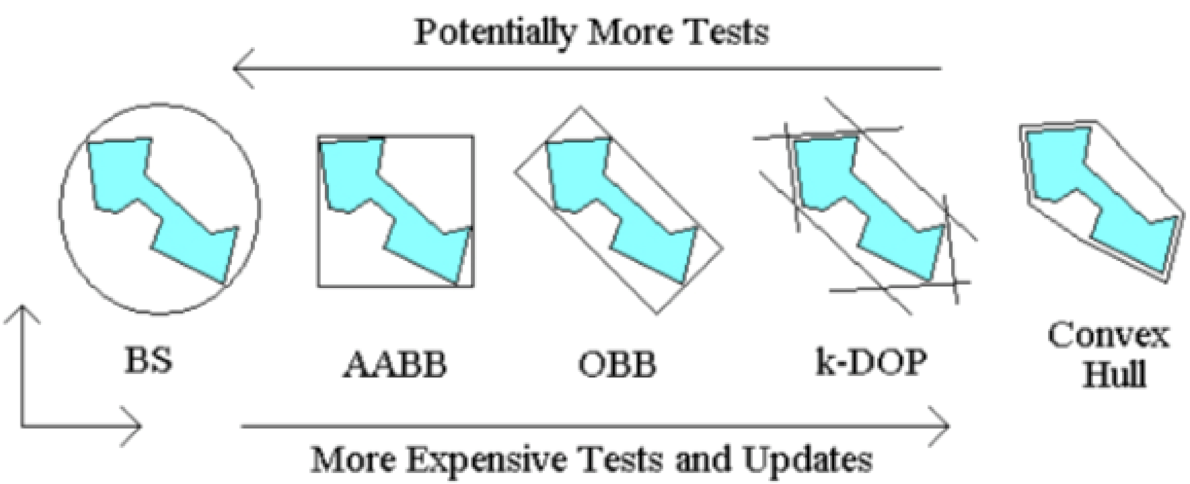
Bounding Volume Hierarchies

- A bounding volume (BV) is a volume that encloses a set of objects.



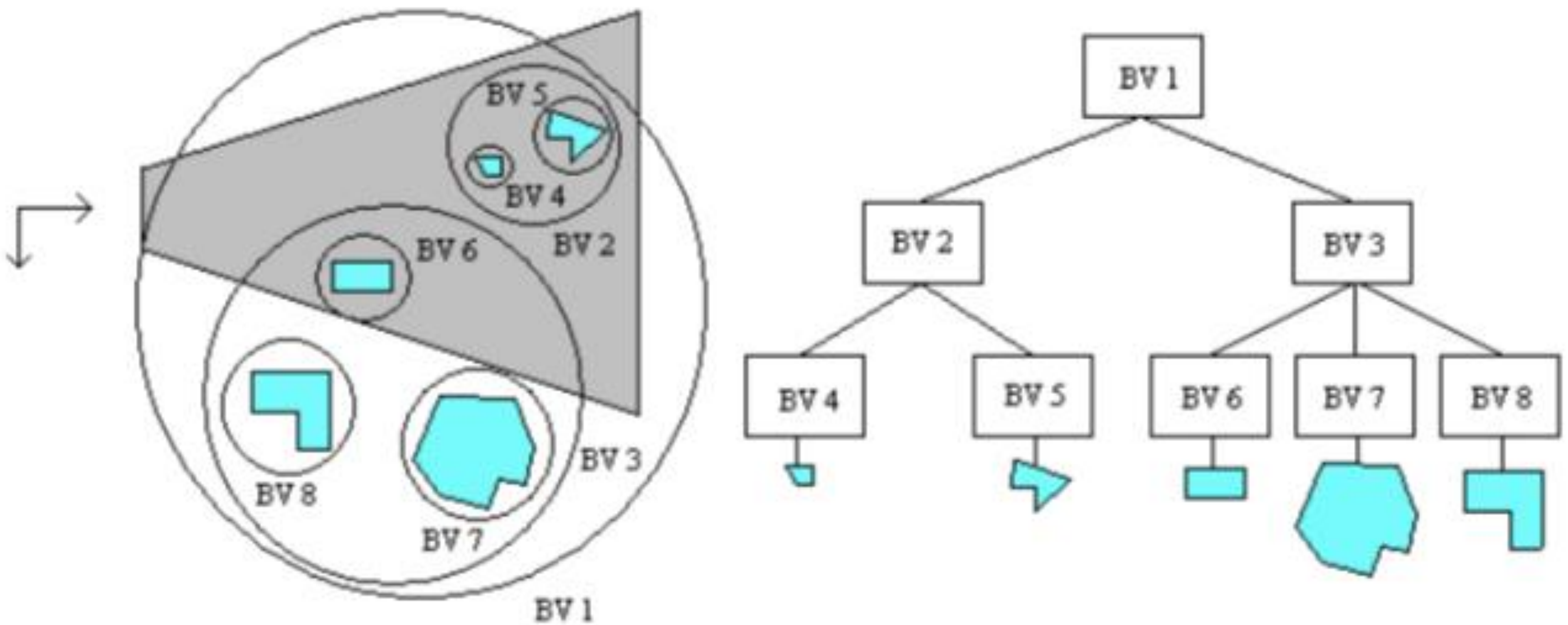
Bounding Volumes Data Structures

- BS – Bounding Sphere
- AABB – Axis Aligned Bounding Box
- OBB – Oriented Bounding Box
- k-DOP – *k* dimensional discrete oriented polytope (a geometric object with "flat" sides)



Bounding Volume Hierarchy

- To reduce tests objects are organized hierarchically



Spatial Partitioning

- **Bounding volume** hierarchies **aimed** to **enclose** and **group** objects **based** on their **spatial relationships**

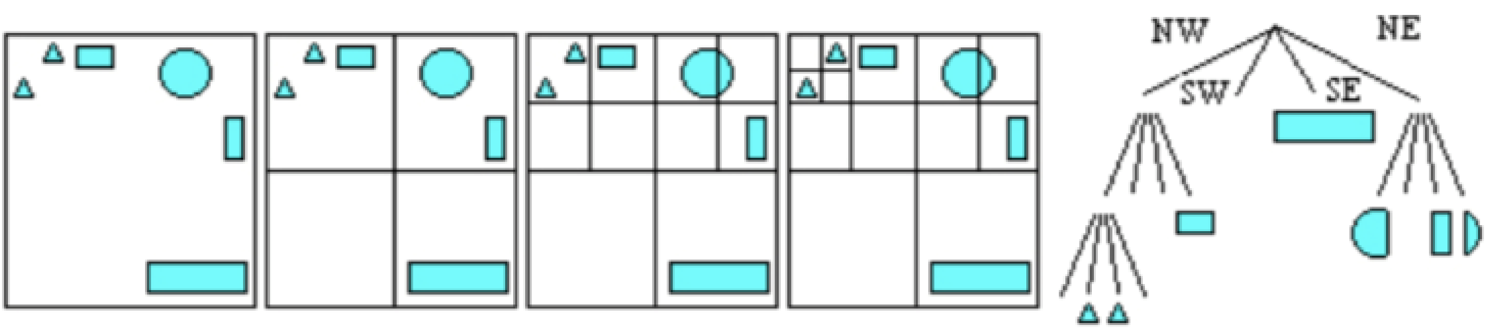
- Space partitioning **divides** a **space** into **non-overlapping regions**, and categorize each according to its contents

Spatial Partitioning Hierarchies

- Quadtrees and Octrees
- KD – Trees
- BSP – Trees

Quadtrees and Octrees

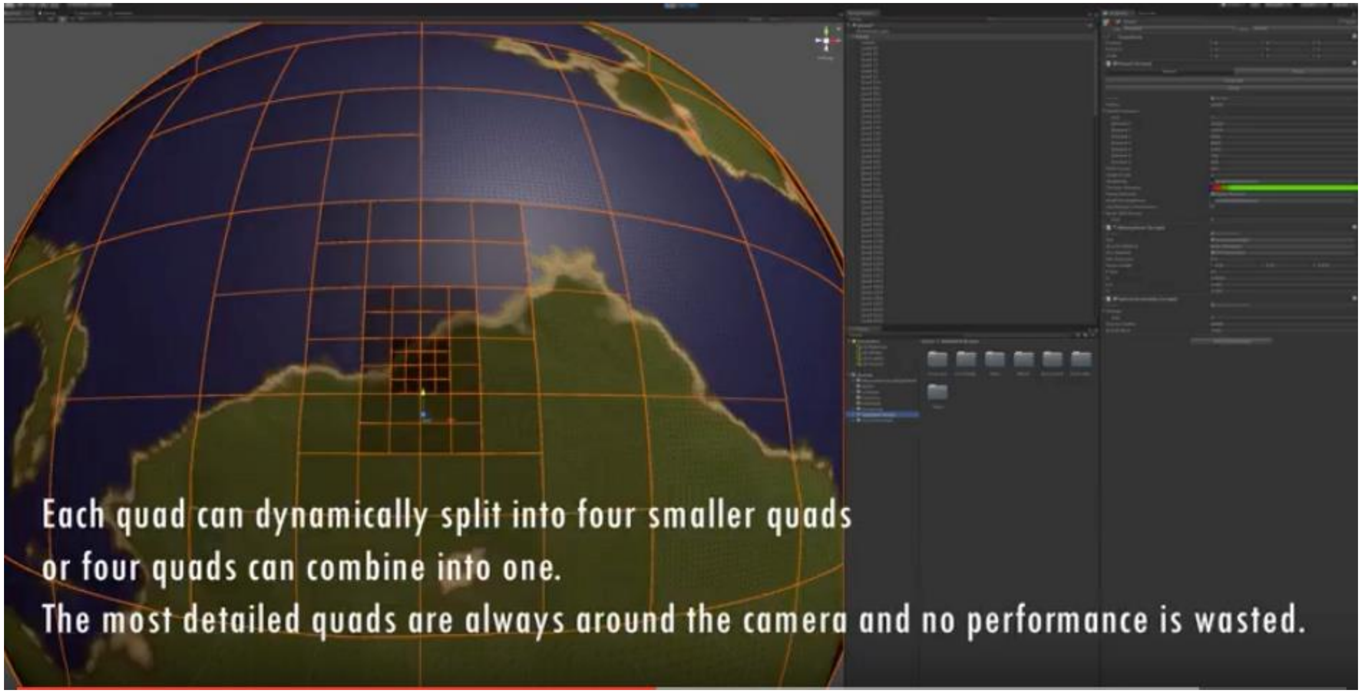
- They hierarchically **divide space** into **equal sized partitions**, by means of iso-oriented hyperplanes
  - Quadtrees in two dimensions
  - Octrees are the three-dimensional variants
- They are usually built using a top-down approach



Constructing a Quadtree

LOD: Quadtrees

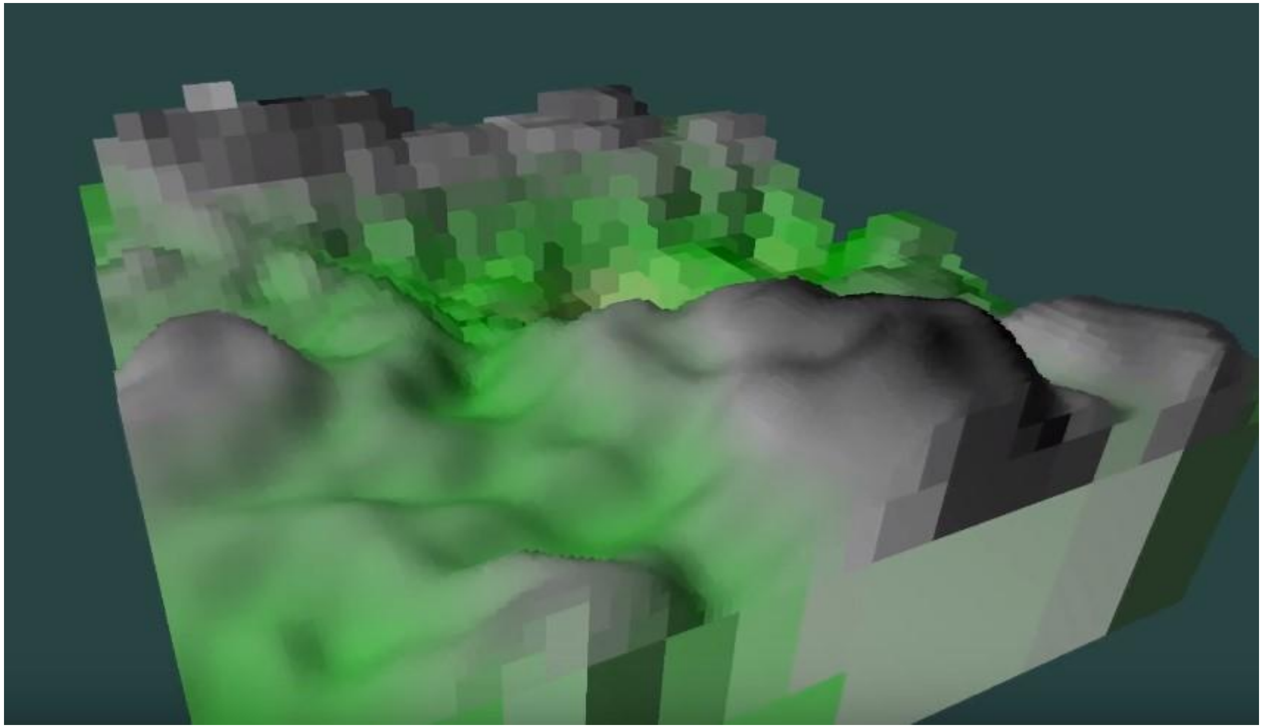
- Different LOD (level-of-detail), higher in closer squares, lower in faraway



- https://www.youtube.com/watch?v=IacfctCKZy8

LOD: Octrees

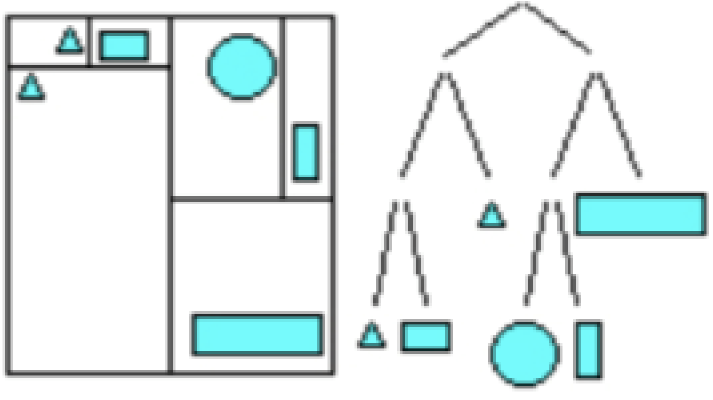
- Different LOD (level-of-detail), higher in closer cubes, lower in faraway



- https://www.youtube.com/watch?v=8ehhxoiWZH0

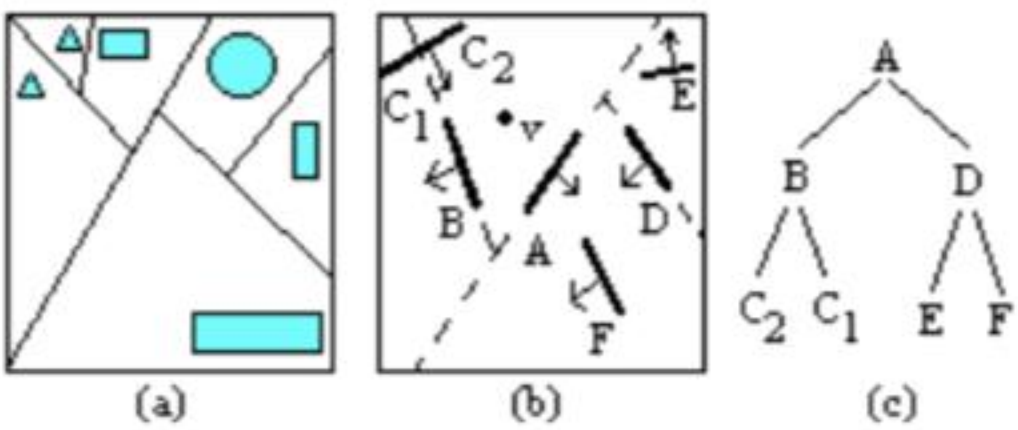
KD-Trees

- **Like Octrees**, k-d trees partition space and enable efficient queries on points
- The hyperplanes are iso-oriented, but **they no longer** have to **divide space regularly**, which allows a certain flexibility in their choice
  - If the object count in any of the new boxes is above a given threshold recursively divide that box
- Key difference: In *d* dimensions a kd-tree is a binary tree that represents a recursive subdivision of the universe into subspaces



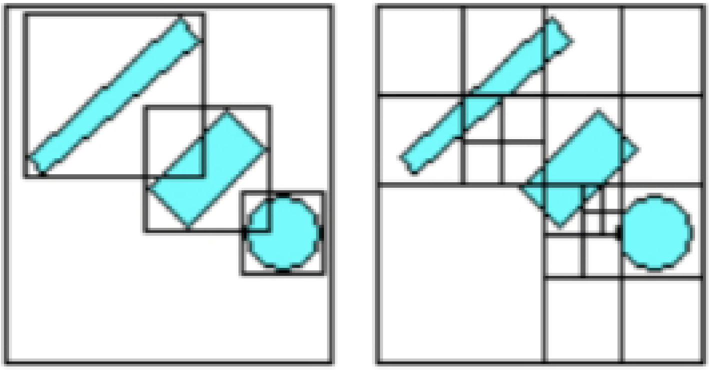
BSP-Trees

- The binary space partitioning tree (BSP-tree) divides the space in dimensional hyperplanes of **arbitrary orientation**
  - Initially one polygon is chosen as the splitter polygon, and stored at the root node
  - Then we have 2 sets of objects (positive and negative sides)



Bounding Volumes vs Spatial Partitioning

- Classifying object vs spaces



Bounding Volumes vs Spatial Partitioning

Bounding Volume Hierarchies	Spatial Partitioning Hierarchies
• Hierarchical Object Representation	• Hierarchical Space Representation
• Object subdivision	• Spatial Subdivision
• Hierarchical clustering of objects	• Hierarchical Clustering of Space
• Classifies Regions of Space Around Objects	• Classifies Objects Around Regions of Space
• Tightly Fits Objects	• Tightly Fills space
• Redundant Spatial Representation	• Redundant Object Representation

- Bounding Volumes: Better for dynamic scenes
- Spatial Partitioning: Better for static scenes
- See description on page 25 [Pires 2001]

Occlusion Culling

Occlusion Culling

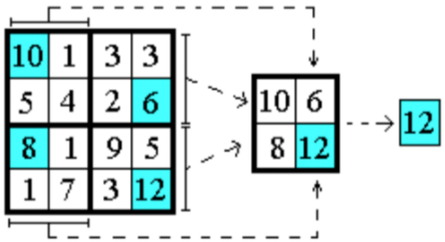
- **Even** if we **render only** the **objects** inside the **view-frustum**
  - some scenes are still too complex to be displayed interactively
  - because there are simply too many objects inside the view-frustum.
- **Occlusion culling techniques** try to determine which objects are visible from a given point (or volume) at a given time, in the presence **of inter-object occlusion**
- Chap 3 [Pires2001]

Occlusion Culling

- Image-Space Techniques
  - Hierarchical Z-Buffer and Hierarchical Visibility
  - Hierarchical Tiling with Coverage Masks
  - Hierarchical Occlusion Maps
- Object-Space Techniques
- Preprocessing Techniques
  - Portals and Cells
  - Cell Subdivision – BSP
  - Ray-Casting
  - Extend Projections

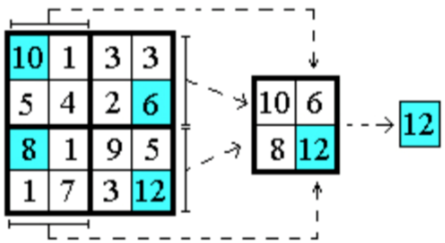
Image Space
Z-Buffer and Hierarchical Visibility

- We have a **z-pyramid** is simply a **set** of **depth buffers** arranged hierarchically.
- The **lowest and finest one**, is an ordinary z-buffer with the **same resolution**, n × m, as the **final image**.



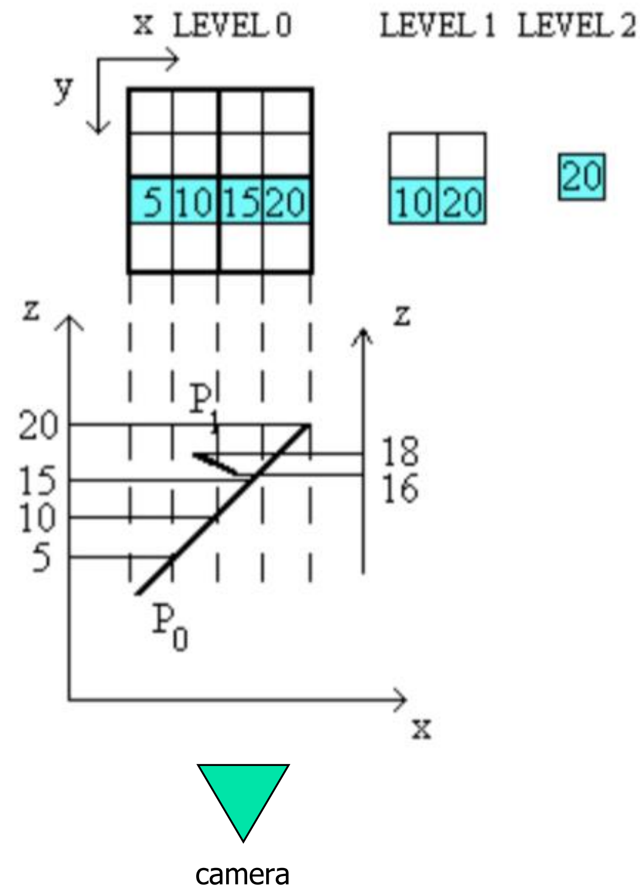
- The second one has a resolution of n/2 × m/2,
- **each entry** stores the **furthest value** of the four corresponding pixels in the first **buffer**.

Z-Buffer and Hierarchical Visibility



Z-pyramid

Z-Pyramid After P_0 and before P_1



Is P_1 occluded by P_0?

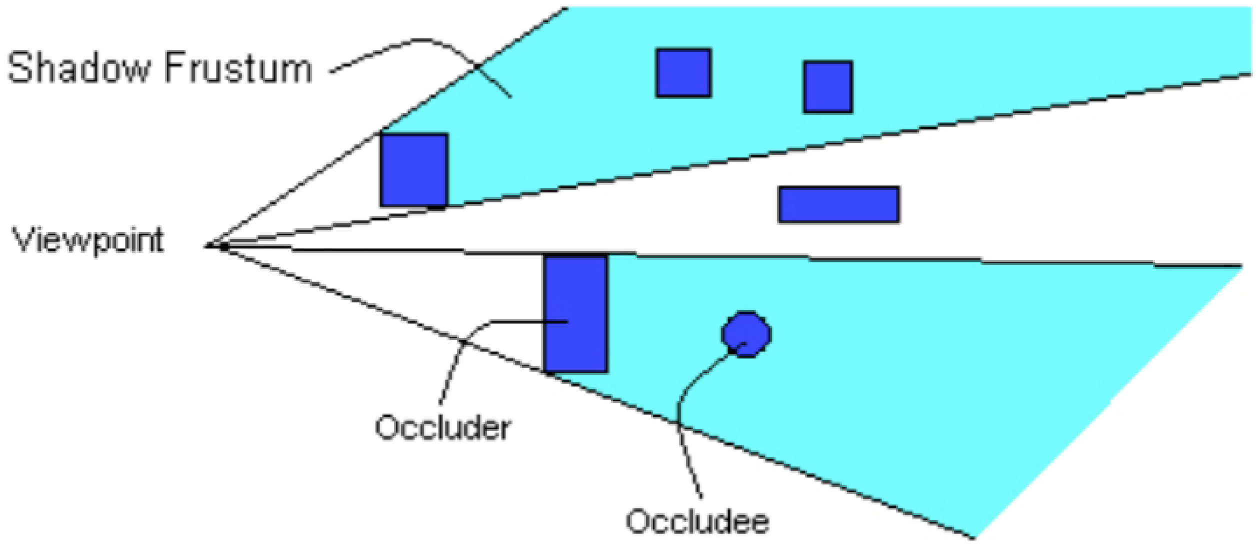
See example page 30

Object-Space Techniques

- Object-Space
  - **Shadow Volume culling**
  - For a given **view point** construct a **shadow volume** for each **occluder**
  - If **an object** is **completely inside** the **shadow volume** than it is not visible and there is **no** need to be **rendered**

Object-Space Techniques

- shadow volume culling

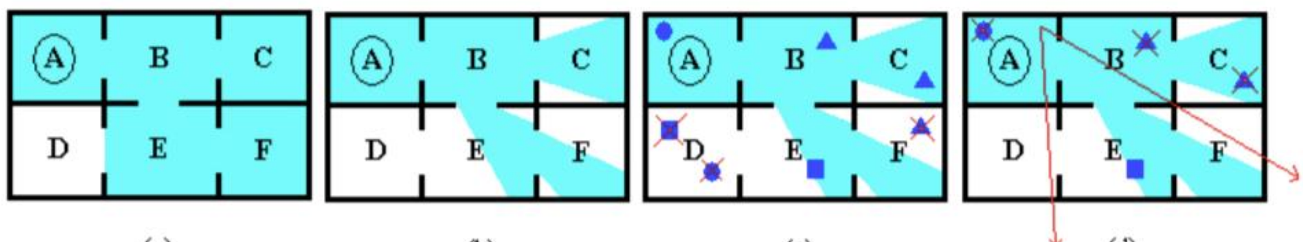


Preprocessing Techniques

- When dealing with **static scenes**, **visibility information** for a given **area/volume** can be **pre-computed**, before the visualization process

Preprocessing Techniques

- Cell and Portal Scenes

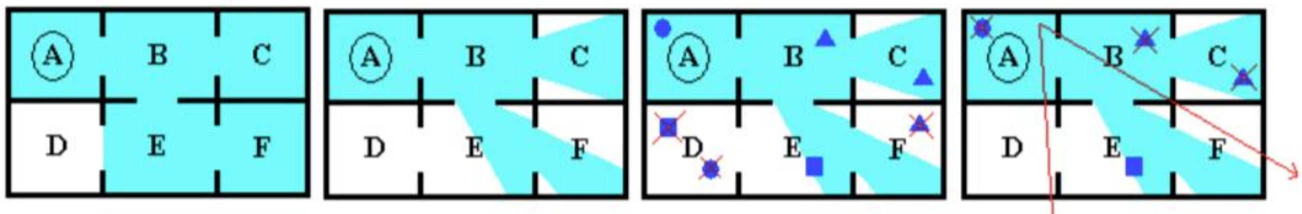


(a) (b) (c) (d)

- Interior architectural scenes exhibit a well-defined structure, that can be divided into a set of closed rooms - cells - that have a set of openings - portals - into other rooms
- **Cell-to-cell visibility:** The set of cells visible from any point in A (fig. a).
- **Cell-to-region visibility:** The visible regions in the visible cells as seen from any point in A (fig. b)
- **Cell-to-object visibility:** The set of all possible static objects visible from all points in A. This constitutes the Possible Visible Set (PVS) for cell A (fig. c).
- **Eye-to-cell visibility:** The set off cells visible from a given point and direction in A (fig. d)

Preprocessing Techniques

- Cell and Portal Scenes



(a) (b) (c) (d)

- BSP Trees Subdivision
  - Using BSP Trees to create an occlusion hierarchy

View-Frustum and Occlusion Culling

- Read Chapters 2 and 3 of:
- Pedro Pires, Dynamic Algorithm Binding for Virtual Walkthrough, 2001