

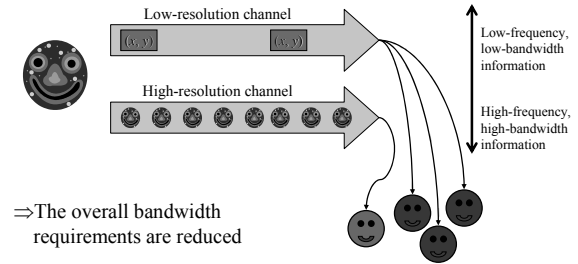
### Exploiting Level-of-Detail Perception

- ◆ Nearby viewers
  - ❖ expect full graphical details
  - ❖ accurate structure, position, orientation
  - ❖ update rate → local frame rate
- ◆ Distant viewers
  - ❖ can tolerate less graphical details
  - ❖ less accurate structure, position, orientation
- ◆ User's focus is typically nearby
- ◆ Many inaccuracies cannot even be detected on a fine-resolution display



### Multiple-Channel Architecture

- ◆ Multiple independent data channels for each entity

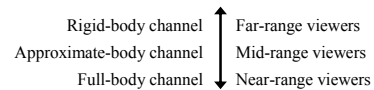


### Implementation Examples

- ◆ Client-server
  - ❖ each transmission identifies its channel
  - ❖ server dispatches data from channels to clients
- ◆ Multicast group for each region
  - ❖ assign multiple addresses for each region
    - one group provides all of the entities' high-resolution channels,
    - another group provides all of the entities' low-resolution channels
- ◆ Multicast group for each entity
  - ❖ assign multiple addresses for each entity
- ◆ Different reliabilities to each channel
  - ❖ low-frequency updates are important
    - lost packets can have a significant impact

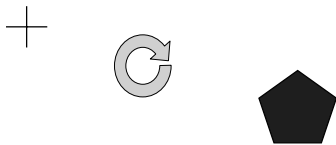
### Selecting the Channels to Provide

- ◆ How many channels to provide for an entity?
  - ❖ more channels: better service for subscribers
  - ❖ each channel imposes a cost (bandwidth and computational)
- ◆ To satisfy the trade-off, three channels for each entity is typically needed
  - ❖ channels provide order-of-magnitude differences in
    - structural and positional accuracy
    - packet rate



### Rigid-Body Channel

- ◆ Demands the least bandwidth and computation
- ◆ Represents the entity as a rigid body
- ◆ Ignores changes in the entity's structure
- ◆ Update types:
  - ❖ position
  - ❖ orientation
  - ❖ structure



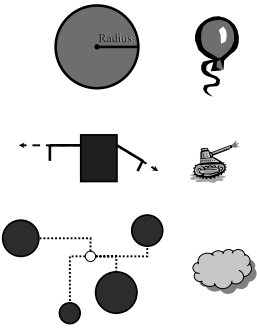
### Approximate-Body Channel

- ◆ More frequent position and orientation updates
- ◆ Hosts can render a rough approximation of the entity's dynamic structure
  - ❖ appendages and other articulated parts
- ◆ Provided information is entity-specific
  - ❖ corresponds to the dominant changes of the structure



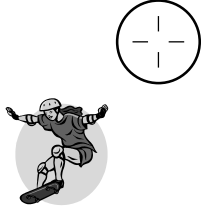
### Common Approximations

- ◆ Radial length
  - ❖ motion towards and away from a centre point
  - ❖ update packets include the current radius
- ◆ Articulation vector
  - ❖ the current direction of the appendage
  - ❖ models a rotating turret, arms and legs
- ◆ Local co-ordinate system points
  - ❖ subset of the entity's significant vertices relative to the entity's local co-ordinate system
  - ❖ the entity is composed of multiple components




### Full-Body Channel

- ◆ Highest level of detail
- ◆ High bandwidth and computational requirements
  - ❖ viewer can subscribe to a limited number of full-body channels
- ◆ Frequent transmissions
- ◆ Position and orientation
- ◆ Accurate structure information



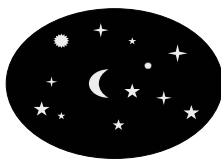
### Exploiting Temporal Perception

- ◆ Render the entity in an accurate location — albeit slightly out-of-date
- ◆ As long as the local user does not interact, small temporal inaccuracies can be allowed
- ◆ Advantages:
  - ❖ works on WANs having great latency
  - ❖ can enhance packet aggregation
  - ❖ can enhance dead reckoning

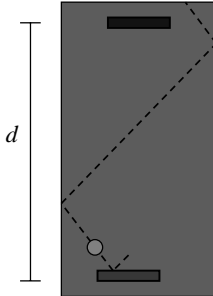


### Active and Passive Entities

<ul style="list-style-type: none"> <li>◆ An active entity                             <ul style="list-style-type: none"> <li>❖ takes actions on its own</li> <li>❖ generates updates</li> <li>❖ human participants, computer-controlled entities</li> <li>❖ cannot be predicted typically</li> <li>❖ rendered using state updates adjusted for the latency</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>◆ A passive entity                             <ul style="list-style-type: none"> <li>❖ reacts to events from the environment, does not generate its own actions</li> <li>❖ inanimate objects (e.g., rocks, balls, books)</li> <li>❖ active entities interact with passive entities</li> <li>❖ rendered according to the latency of its nearest active entity</li> <li>❖ reacts instantaneously to the actions of a nearby active entity</li> </ul> </li> </ul>
---	--

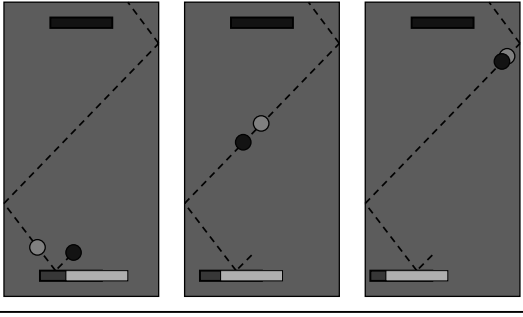


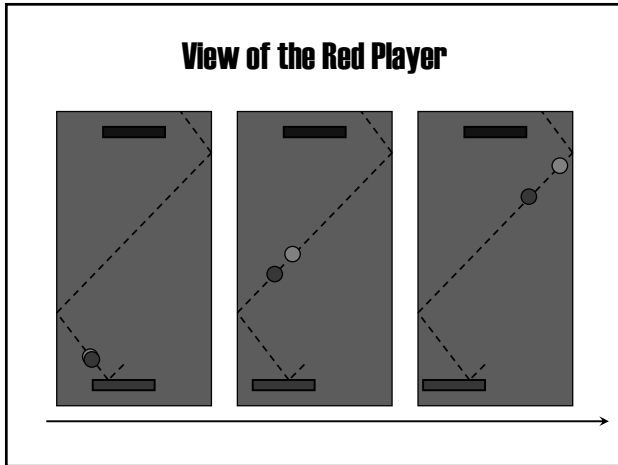
### Example: Pong



- ◆ Two active entities: paddles
  - ❖ movement unpredictable
- ◆ One passive entity: ball
  - ❖ movement predictable
- ◆ Latency of  $d$  seconds

### View of the Blue Player





### Pong: A Summary

- ◆ Each player sees a different representation of the same playing field
- ◆ The ball accelerates as it approaches the local player's paddle
- ◆ The ball decelerates as it approaches the remote player's paddle
- ◆ The ball's rendered position alternates between
  - ❖ the current time
    - meaningful interaction for local player
  - ❖ a past time reference
    - network latency
    - observing meaningful interaction for remote player

### 3½-Dimensional Playing Field

- ◆ Represent each player's perception as a four-dimensional co-ordinate system  $(x, y, z, t)$ 
  - ❖  $x, y, z$ : the spatial position relative to the local player's current position
    - local player at  $(0, 0, 0)$
  - ❖  $t$ : the time associated with rendered information from that position
    - local player rendered at current time:  $t = 0$
    - opposing player:  $t = -d$

### Co-ordinate Systems

Blue Player

Red Player

### Properties of the Co-ordinate System

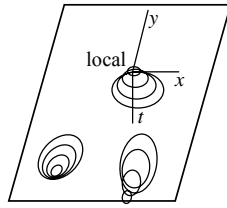
- ◆ The co-ordinate system is defined independently for each player
- ◆ Depends on the player's current position and the delay of arriving information
- ◆ Changes dynamically as the player moves or as the network properties change
- ◆ Defines how a passive object should be rendered
- ◆ Two interacting objects are rendered at the same time reference point
- ◆ Each user perceives all collisions correctly
- ◆ Objects that approach the local user are rendered in the user's time
- ◆ Smooth movement

### Generalizing the Local Temporal Contour

- ◆ Limitations:
  - ❖ players are capable of moving along a single axis only
  - ❖ supports two active objects only
- ◆ Generalization to a 4D co-ordinate system requires preserving for the local user:
  - ❖ interacting naturally with passive objects in vicinity
  - ❖ seeing remote interactions (passive-to-passive, passive-to-active) naturally
  - ❖ perceiving smooth motion of remote objects

### Local Temporal Contour

- ◆ The local user at (0, 0, 0)
- ◆ Each active object is assigned a  $t$  value corresponding to its latency
- ◆ Interpolate the contour over all active objects including local
- ◆ Contour defines a suitable  $t$  value for each spatial point



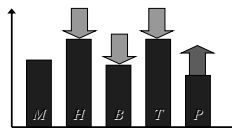
### Limitations

- ◆ Varying latency can cause entities to (unnaturally) jump forward or backward in time
  - ❖ use averaged latency to dampen the effect
- ◆ What if an update packet is delayed considerably?
  - ❖ predict entity's past position, dead reckoning
- ◆ Computational requirements
  - ❖ compute the contour using only the nearest active entities

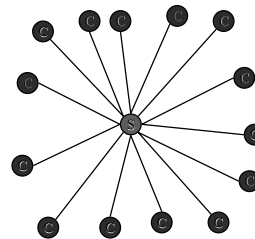


### §6.4 Enhancing the System Architecture

- ◆ Change the network software architecture
- ◆ Basic structures: client-server and peer-to-peer
- ◆ Augment and combine basic structures
  - ❖ server clusters
    - partition clients across multiple servers
    - partition the NVE across multiple servers
  - ❖ server hierarchies
  - ❖ peer-server systems



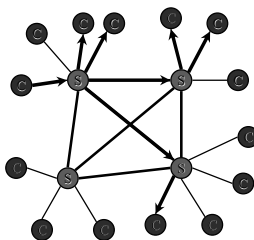
### Traditional Client-Server



- ◆ Server may act as
  - ❖ broadcast reflector
  - ❖ filtering reflector
  - ❖ packet aggregation server
- ◆ Scalability problems
  - ❖ all traffic goes through the server

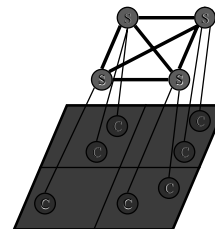
⇒ Server clusters

### Partitioning Clients across Multiple Servers



- ◆ The servers exchange control messages among themselves
  - ❖ inform the interests of their clients
- ◆ Reduces the workload on each server
- ◆ Incurs a greater latency
- ◆ The total processing and bandwidth requirements are greater

### Partitioning the NVE across Multiple Servers



- ◆ Each server manages clients located within a certain region
- ◆ Client communicates with different servers as it moves
- ◆ Eliminates a lot of network traffic
- ◆ Requires advanced configuration
- ◆ Is a region visible from another region?
- ◆ Aggregation servers are a special case of NVE server partitioning

