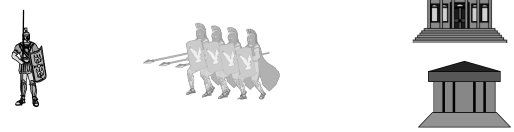
§9.4 Local Perception Filters

- exploiting human's perceptual limitations
  - level-of-detail: less details where they cannot be observed
  - image, video and audio compression
- local perception filters
  - exploits temporal perception
  - shows possibly out-of-date information (≠ dead reckoning)
  - ensures consistent interaction
  - allows to introduce artificial delays (e.g., bullet time)

Exploiting Perceptual Limitations

- Humans have inherent perceptual limitations



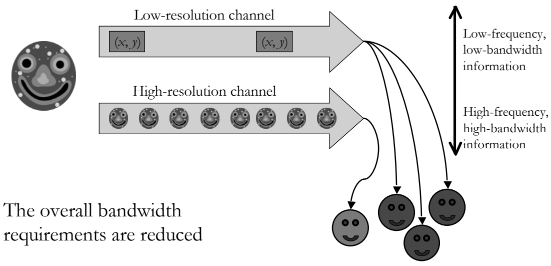
Two approaches to exploit
1. Information can provided at multiple levels of detail and at different update rates
2. Mask the timeliness characteristics of information

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



The overall bandwidth requirements are reduced

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

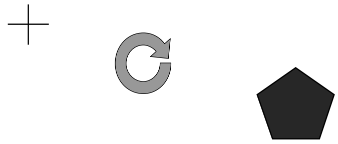
Selecting the Channels to Provide

- 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		Far-range viewers
Approximate-body channel		Mid-range viewers
Full-body channel		Near-range viewers

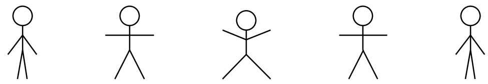
Rigid-Body Channel

- 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



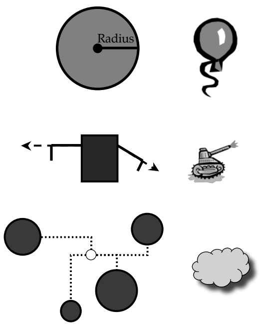
Approximate-Body Channel

- 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



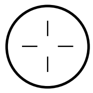
Common Approximations

- 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

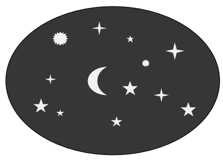


Local Perception Filters (LPFs)

- introduced by Sharkey, Ryan & Roberts (1998)
- a method for hiding communication delays in networked virtual environments
- exploits the human perceptual limitations by rendering entities slightly out-of-date locations based on the underlying network delays
  - causality of events is preserved
  - rendered view may have temporal distortions
  - rendered view ≠ real view

Active and Passive Entities

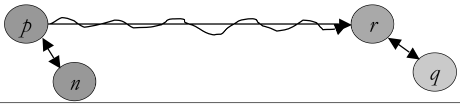
- An active entity (i.e., player)
  - takes actions on its own
  - generates updates
  - human participants, computer-controlled entities
  - cannot be predicted typically
  - rendered using state updates adjusted for the latency



- A passive entity
  - reacts to events from the environment, does not generate its own actions
  - inanimate objects (e.g., rocks, balls, books)
  - active entities interact with passive entities
  - rendered according to the latency of its nearest active entity
  - reacts instantaneously to the actions of a nearby active entity

Rules of LPFs

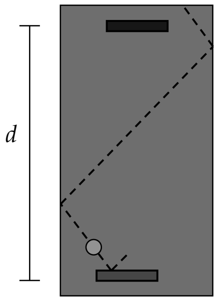
1. Player should be able to interact in real-time with the nearby entities.
2. Player should be able to view remote interactions in real-time, although they can be out-of-date.
3. Temporal distortions in the player's perception should be as unnoticeable as possible.



Interaction Between Players

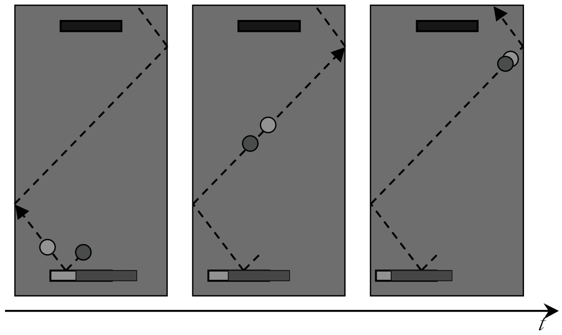
- interaction = communication between the players
  - local players: immediate
  - remote players: subject to the network latency
    - time frame = current time – communication delay
- interaction = players exchanging passive entities
  - passive entities are predictable ⇒ they can be rendered in the past (or in the future)
- a passive entity can change its time frame dynamically
  - the nearer to a local player, the closer it is rendered to the current time
  - the nearer to a remote player, the closer it is rendered to its time frame

Example: Pong

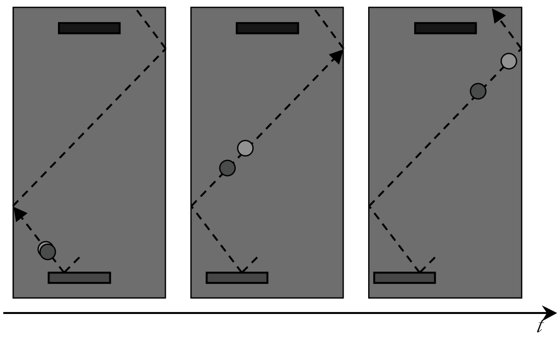


- Two active entities: paddles
  - movement unpredictable
- One passive entity: ball
  - movement predictable
- Latency of *d* seconds

The View of the Blue Player

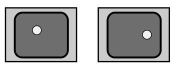


The View of the Red Player



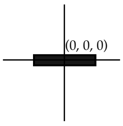
Pong: A Summary

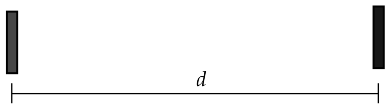
- Each player sees a different representation of the same playing field
- The ball accelerates as it approaches the local player's paddle
- The ball decelerates as it approaches the remote player's paddle
- The ball's rendered position alternates between
  - the current time
    - meaningful interaction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



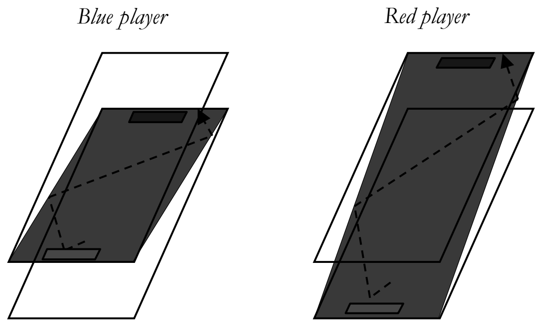
3½-Dimensional Temporal Contour

- 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: $t = -d$

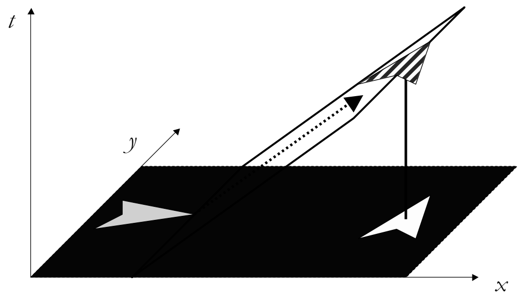




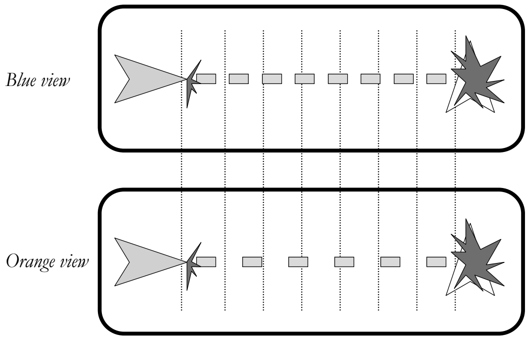
Temporal Contours in Pong



Temporal Contour (from the Blue Player's Perspective)

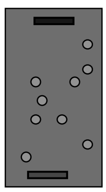


Temporal Distortion



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

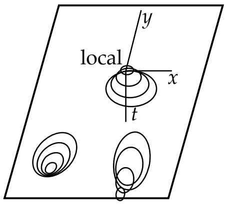


Generalizing the Local Temporal Contour

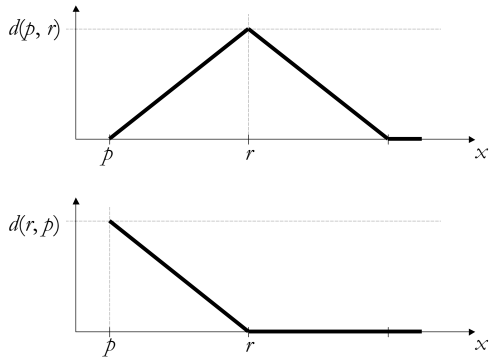
- Limitations:
  - players are capable of moving along a single axis only
  - supports two active objects only
- Generalization to a 4D co-ordinate system requires preserving for the local user:
  - interacting naturally with passive objects in vicinity
  - seeing remote interactions (passive-to-passive,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

Local Temporal Contour

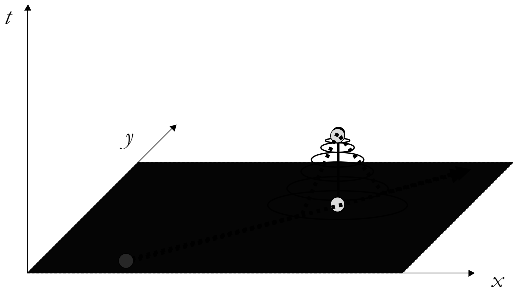
- The local user at (0, 0, 0)
- Each active object is assigned a t value corresponding to its latency
- Interpolate the contour over all active objects including local
- Contour defines a suitable t value for each spatial point



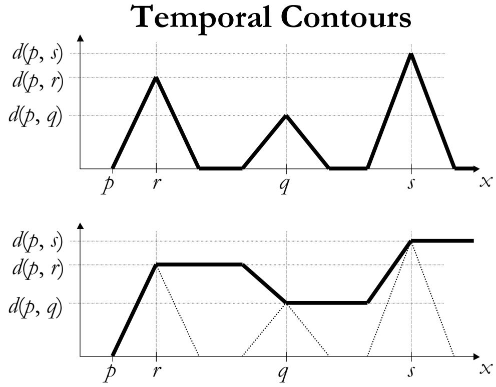
Linear Temporal Contours



2½-Dimensional Temporal Contour



Multiple Players: Aggregating the Temporal Contours



Worth Noting

- simple linear functions instead of continuous temporal contours
- LPFs are the 'opposite' of dead reckoning
  - no prediction for remote players
- the closer the players get, the more noticeable the temporal distortion becomes
  - in critical proximity interaction becomes impossible
  - no mêlée


Problems

- possibly visual disruptions on impact ⇒ shadows (see the lecture notes for details)
- sudden changes in the player's position or delay can cause unwanted effects
  - if a player leaves the game, what happens to the temporal contour?
  - third party instrusion: someone with a high delay 'blocks' the incoming entities
  - jitter: entities start to bounce back and forth in time