Distributed Collision Detection

Tom Chen
Project Presentation: Comp522
School of Computer Science
McGill University
December 13, 2008
Online Multiplayer Games

- Multiple stations connected through network
- Master objects
  - Controlled by player’s own station.
- Duplicate objects
  - Copies of master object that reside on other stations.
  - Receive master’s position through network connections.
Challenges

• Network Latency
  – Position updates from master arrived late.
  – Packet arrival rate is asynchronous with game’s frame rate.
  – Different latency between stations affects consistency between stations

• Bandwidth
  – Frequency of updates.
  – Size of packets.
Dead Reckoning

• Extrapolation
  – Packet arrived at $t_0$, with $p_0$, $v_0$ and $a_0$
  – Extrapolate $p_1$, $v_1$ and $a_1$ at given frame time
Network Latency

Station 1

Station 2

Low Latency

High Latency

High Latency with Dead Reckoning

Station 3

Station 4
Position History-Based Dead Reckoning

• PHBDR
  – Send position only.
  – Store received position in a history list.
  – Extrapolate using history positions.
  – Reduced bandwidth usage
Position History-Based Dead Reckoning

• First order extrapolation
  – Extrapolate $p_1$ using $p_0$ and $p_{-1}$
  – Good extrapolation for linear motion
Position History-Based Dead Reckoning

• Second order extrapolation
  – Extrapolate $p_1$ using $p_0$, $p_{-1}$, $p_{-2}$
  – Smooth extrapolation for non-linear motion
High Network Latency

- Each extrapolation create a small error
- Network latency are bursty and if 10 packets are late...
High Network Latency

• Extrapolation based on extrapolation based on extrapolation based on extrapolation based on extrapolation based on extrapolation based on extrapolation based on extrapolation based on extrapolation based on extrapolation based on extrapolation based on extrapolation ...
Non-Linear Motions

- Objects that are moving non-linearly are subject to higher error during high network latency.
Collision Detections

• With high latency during collision
  – Dead reckoning extrapolate the wrong position
  – Penetration
  – Inconsistency game state between stations
Local Correction

• During collisions, allow duplica to resolve collision locally.
• After collisions, master regain control of duplica.
Problem: False Collision

• Situation
  – High network latency
  – Local correction is on
  – Objects moving non-linearly with large errors...
False Collision
False Collision
False Collision
False Collision
False Collision
False Collision
Inconsistency
Motivation

• True Collision vs. False Collision
  – How to detect false collision?
  – When to turn on/off Local Correction?
• Ultimately, consistency among stations
  – All or none detects the collision
Proposal

• Observe Non-linear motions
  – Relationship between latency and distance error
Modeling and Simulation

• Build a online multiplayer game system model
• Allow fast prototyping with different strategies
• For the Comp 522 project:
  – Explore the Continuous Spatial distribution
  – Measure latency vs. distance error
Modeling and Simulation

- DEVS + Continuous Spatial Distribution
  - DEVS models the network and stations
  - Each station has a large 2D continuous spatial state that models the game
  - Games consist of objects that moves at continuous time
  - DEVS samples the game state at discrete time and output to graphic interface
Continuous Spatial Distribution

• Game objects
  – Continuous state in 2D space, \((x, y)\)
  – Moves from one state to the other under equations of motion in continuous time

\[
\begin{bmatrix}
\text{\[ \bullet \rightarrow \bullet \]} \\
\end{bmatrix} = \begin{array}{c}
x = (x, y) \\
v = dx/dt \\
a = dv/dt
\end{array}
\]
Continuous time vs. Discrete Time

• Difficult to model continuous time

• Abstraction: Discretize time
  – Break time into discrete segments, $\Delta t$
  – Like what we did in Causal Block Diagram assignment
  – Size of $\Delta t$ is the frame rate of the game
  – Smaller $\Delta t$, smoother animation
Continuous time vs. Discrete Time

![Diagram showing continuous and discrete time](image)
Continuous State

• If observe the state between each frame, objects may have different position
  – E.g. $x$ is continuous
DEVS + Continuous Spatial Distribution

• Combine DEVS with Continuous spatial Distribution
  – Time advance $ta() = \Delta t$
  – Internal transition updates all game objects
  – Output function renders the objects
Simulator

Station

Game

Main Loop

Network

Connection

Connection

Station

Game

Main Loop

Monitor
Experiments

• Latency vs. average distance error
  – An object moves in circular motion
  – Measure distance error between Master and Duplica
  – Increase latency for each experiment
  – 10 seconds per experiment

• Expect a exponential growth of distance error
  – Duplica spiral out due to accumulation of errors
Latency

• Frame rate
  – 50 fps = 20ms per frame

• Latency
  – Zero latency if packets arrived at 0 to 20ms per packet.
  – Packet arrival rate from 21ms or greater will be out of sync with the frame rate
22ms/Packet
23ms/Packet
24ms/Packet
26ms/Packet
28ms/Packet
35ms/Packet
40ms/Packet
Average Distance Error

![Graph showing Average Distance Error vs Latency (ms). The graph displays a linear increase in average distance error with increasing latency.]
Average Distance Error

• Rate of change in average distance error decreases around 24 - 25ms per packet??
  – Need to investigate

• Possibilities
  – Buffer size
  – Incorrect network model
Conclusion

• Inconsistency in distributed collision detection is due to network latency.
• Discretize time to combine Continuous Spatial Distribution and DEVS.
• Unexpected average distance error growth
Future Work

• Provide mathematical relation between latency and average distance error.
• Improve the model.
• Implement different strategies to detect false collisions.