

# **Data Management for Visualizing Large Virtual Environments**

*Tsai-Yen Li, and Chih-Wei Chiang*

Computer Science Department  
National Chengchi University  
Taipei, Taiwan, R.O.C.  
Email: {li, s8409}@cs.nccu.edu.tw

ISMIP' 99

Dec, 1999

## **Outline of the Talk**

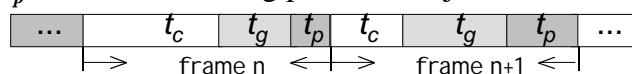
- **Introduction**
- **Problem descriptions and related work**
- **Proposed approach**
  - visibility preprocessing
  - prioritized object prefetching
  - hybrid caching model
- **System architecture and implementation**
- **Experimental results**
- **Conclusion and future work**

## Introduction

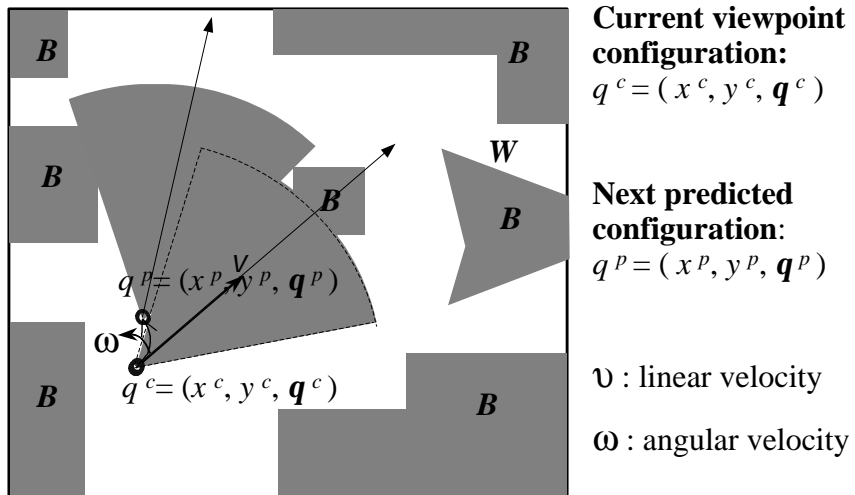
- **Motivation:** problems of interactive navigation in a large virtual environment
  - Physical memory is never enough for large virtual worlds.
  - Delay and discontinuity in scene changes is disturbing.
- **Intuitions: effective data management**
  - on-demand incremental data retrieval
  - distributing object loading smoothly in each frame
- **Proposed techniques:**
  - **visibility precomputation**
  - **object prefetching**
  - **hybrid cache model**

## Problem Description

- **World description:** geometric models and sizes of the objects in the world are stored in a database, and a 2D layout of the virtual world is available.
- **Limited physical memory:** size of a 3D virtual world does not fit into available physical memory.
- **Time utilization for each frame update:** perform object retrieval and graphics rendering in each frame cycle.
  - $t_c$ : time for retrieving objects in the current frame
  - $t_g$ : time for graphics rendering
  - $t_p$ : time for retrieving predicted objects



## View Model and Notations



## Related Work

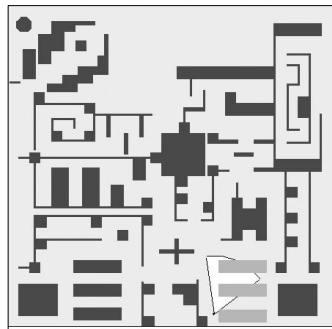
- **Interactive building walkthrough:**
  - Data management scheme : [Airey 90], [Funkhouser 92], [Teller 91], etc.
- **Scalability for a large number of clients:**
  - Reducing communication complexity: [Carlsson 93], [Li 99], [Macedonian 95], etc.
- **Cache model:**
  - Least-Recently-Used(LRU) policy: [Franklin 92], etc.
  - Most Required Models(MRM) : [Chim 98]
- **Prefetching strategy:**
  - One-step prefetching: [Chim 98], [Teller 91], etc.

## Our Approach

- **Visibility pre-computation:**
  - For a given world description, precompute distinct visible sets for all possible discrete viewpoint configurations.
  - Use the visible set to quickly identify the set of objects that need to be retrieved at run time
- **Prioritized object prefetching:**
  - In each time frame, prefetch the most relevant objects of a viewpoint configuration for the near future
- **Hybrid cache model:**
  - Use both access time and object relevance to account for both temporal and spatial localities
  - consistent with the above prefetching strategy.

## Visibility Pre-computation

- **Precompute distinct visible sets off-line for quickly identifying visible objects at run time**

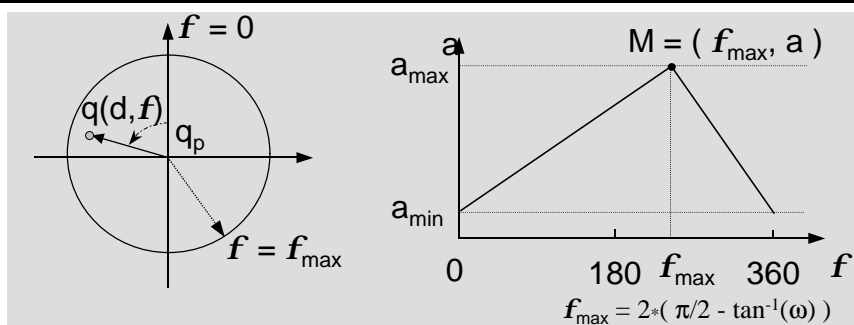


- **Adopt a more conservative approach:**
  - retrieving objects in the union of visible sets for all of the current configuration's 1-neighbors

## Object Prefetching: Prioritized Spatial Prefetching

- **Using a n-by-n relevance matrix to define the relevance values of neighboring configurations:**
  - analytically defining relevance values according to distance from viewpoint, viewing direction, and velocity (linear and angular)
- **Retrieving objects according to the order of their relevance to the predicted configuration:**
  - visiting the configurations in the matrix in the order of their relevance values
  - prefetching objects in these configurations within available time

## Parameters for Computing a Relevance Matrix



Relevance value( $w$ ):  $w(a, d) = -ad^2 + c$ , ( $c$ : constant)

$$a = f(f) = a_{\min} + \frac{(a_{\max} - a_{\min})}{f_{\max}} * f \quad , \text{ if } f < f_{\max}$$

$$a_{\max} + \frac{(a_{\max} - a_{\min})}{(f_{\max} - 2\pi)} * f \quad , \text{ if } f > f_{\max}$$

$$f_{\max} = \pi - 2 * \tan^{-1}(\omega)$$

$$\begin{matrix} v & \longrightarrow & f_{\max} & \longrightarrow & a(f) & \longrightarrow & w(a, d) \\ \omega & \longrightarrow & a & & & & \end{matrix}$$

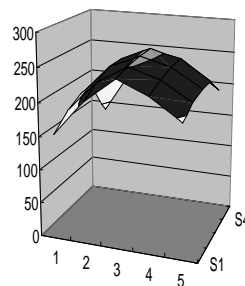
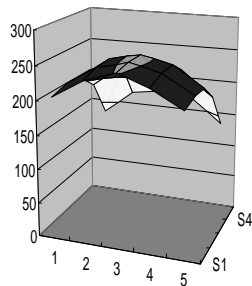
## Examples of Computed Relevance Matrices

209	235	251	235	209
216	243	254	243	216
213	244	255	244	213
188	224	235	224	188
133	169	175	169	133

157	209	246	225	195
177	224	251	235	209
188	235	254	243	216
181	236	255	244	213
136	205	235	224	188

$\nu = 0, \omega = 0$

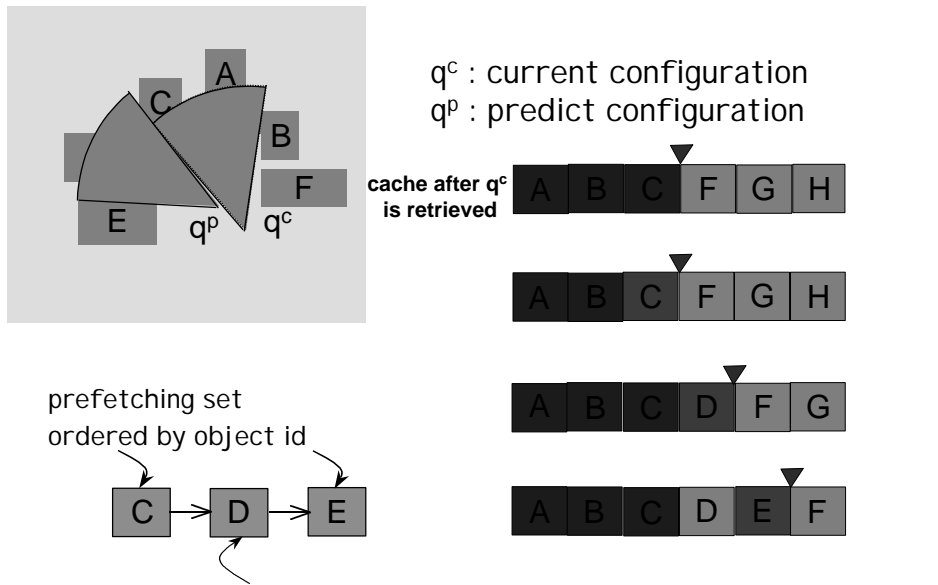
$\nu = 0.5, \omega = -0.5$



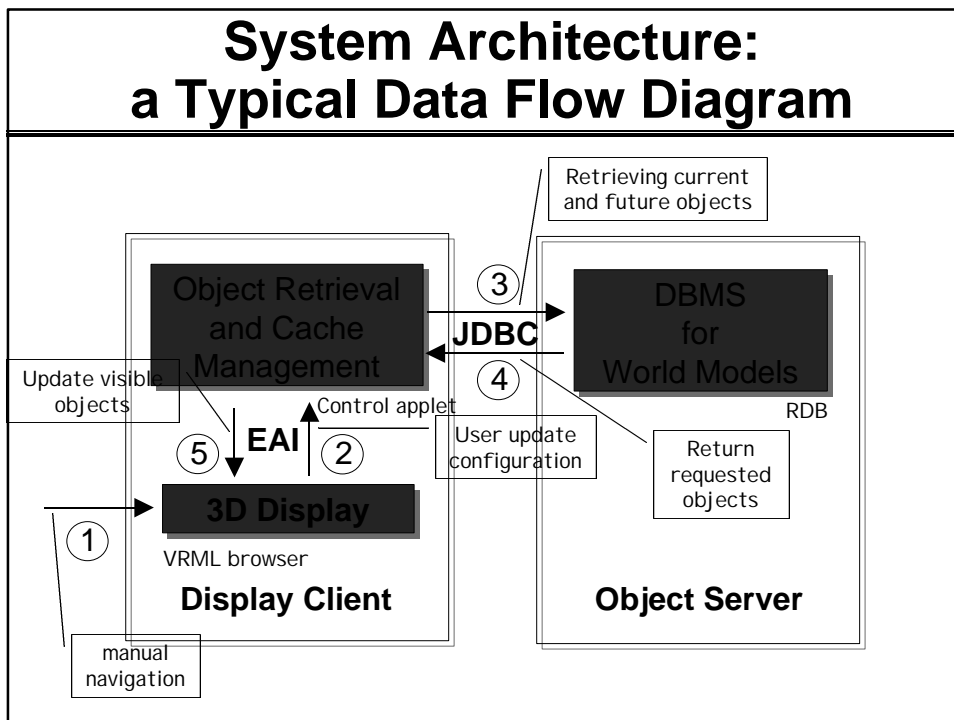
## Data Caching: Hybrid Cache Model

- **Taking advantage of both temporal and spatial localities:**
  - For objects retrieved at different frames: use traditional LRU.
  - For objects retrieved at the same frame: order the replacement queue by their relevance values.
- **The model is consistent with the prefetching strategies we used while keeping the advantages of LRU.**

## Example of How Hybrid Caching Works



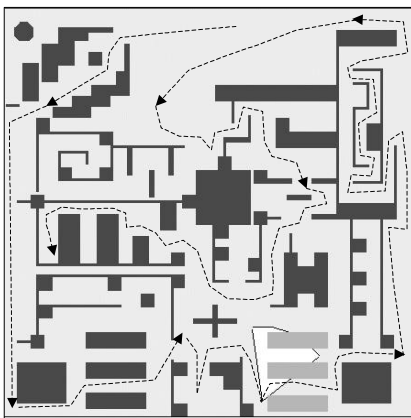
## System Architecture: a Typical Data Flow Diagram



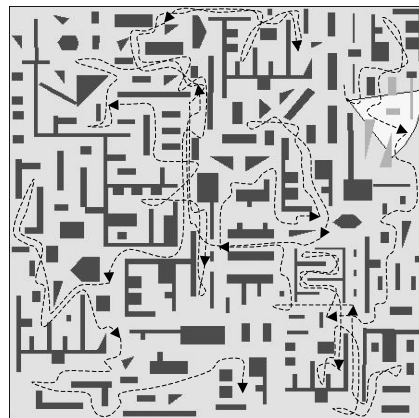
## Implementation and Experiments

- All modules except for the VRML browser are implemented in Java.
- VRML browser  $\leftrightarrow$  Control applet:
  - External Authoring Interface (EAI)
- Control applet  $\leftrightarrow$  DBMS
  - JDBC
- Experimental Path
  - Collision-free navigation paths are generated by a path planner
- Experimental platform:
  - regular PC with a Pentium II 300 MHz processor running Windows NT

## Examples of Experimental Environments

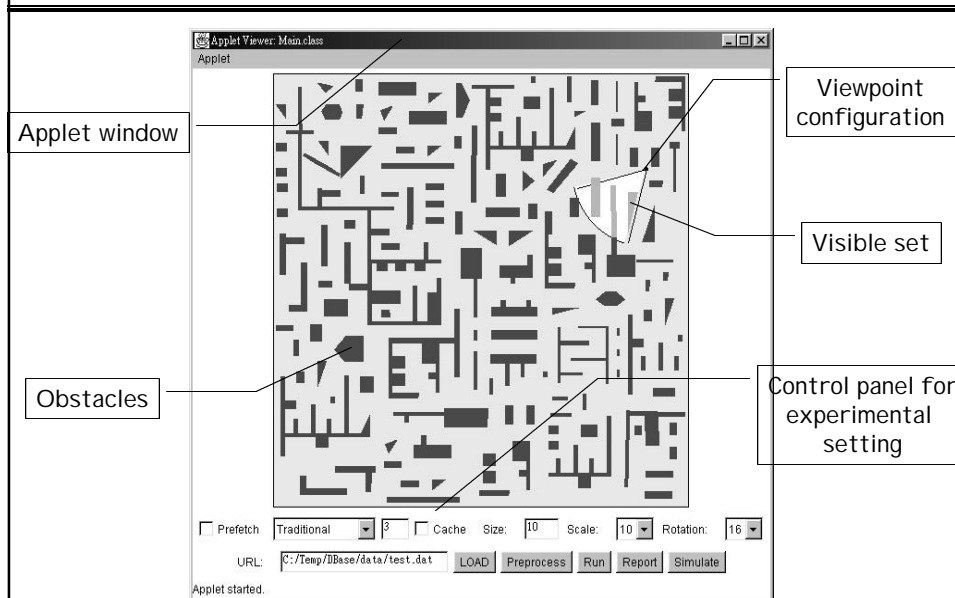


**Example 1:**  
128 objects, 1068 steps



**Example 2:**  
370 objects, 1780 steps

# Graphical User Interface for Simulation



## Experimental Results: Improvement of display smoothness

- **Experiment data of the second example:**

	case 1	case 2	case 3	case 4
<b>AVRG (ms)</b>	305.6	97.2	97.7	103.8
<b>STD (ms)</b>	133.6	52.6	51.8	46
<b>Hit Ratio (%)</b>	N/A	90.9	90.8	95.2

case 1: no caching/prefetching  
 case 2: LRU caching only  
 case 3: w/ caching and one-step prefetching  
 case 4: w/ caching and relevance mask prefetching

- **Improvement:**

- **LRU cache + no prefetching:**
  - ✓ AVRG: 214%, STD: 154% (case 2/case 1)
- **LRU cache + one-step prefetching:**
  - ✓ slightly improved (case 3/case 2)
- **Hybrid caching + relevance-matrix prefetching:**
  - ✓ STD was further improved to 190% (case 4/case 1)

## Conclusion and Future Work

- **Proposed an effective data management scheme utilizing**
  - discrete visibility precomputation
  - hybrid cache replacement policy
  - prioritized object prefetching**to achieve interactive visualization of large virtual worlds.**
- **Future extensions:**
  - **completing system integration and doing more experiments**
  - **finding out the effects between available cache size and allowable prefetching time**
  - **adopting adaptive prefetching time**

**Q & A**