Visorama

Luiz Velho

IMPA - Instituto de Matemática Pura e Aplicada

Outline

1. Context: *A Little History*
2. Visorama: *What is IT?*
3. Initial Development: *Real Time Panoramas*
4. Fundamentals: *General Concepts*
5. Applications: *Suggestions of Projects*
6. Software: *Basic Library*
7. Links: *Related Work*
8. Future: *Ideas*

1 – Context: *A Little History*

- Real Panoramas
- Virtual Panoramas
- First Generation Systems

Real Panoramas

- Paintings
- Photographs

Panorama Installations I

Panorama Installations II
Virtual Panoramas

- Mathematical Concept
  - Model
  - Visualization
- Computer Implementation
  - Hardware
  - Software
- Applications
  - Multimedia
  - Virtual Reality

Model of Panorama

Viewing a Panorama

- Camera Transformation

Current Panorama Systems

- Characteristics
  - Small Images, Embedded in Applications
  - Interactive, but Not Real-Time
  - Low Performance, Consumer Hardware
- Formats and Companies
  - QuickTime VR (Apple)
  - Real Space (Live Picture)
  - PhotoBubble (Omniview)
  - Surround Video (IBM)

Hardware

- Networked Personal Computer

Software

- Multimedia (CD):
  - Director
- Web Applications
  - Netscape
Navigation

- Hyperlinks (Hot Spots)
  - Moving Between Nodes
  - Activate another Application

Production Tools

- Creation and Authoring

2 - Visorama: What is IT?

Definition:

“A Virtual Reality System based on Panoramas*”

Visorama: A New Concept

- Virtual Reality based on Panoramas
  - Visual-Realism*
  - Immersive
  - Natural Interaction
- Complete System
  - Dedicated Hardware
  - Viewing Software*
  - Authoring Environment*

Equipment Design

- Binocular device
- Stereo Sound

- Degrees of freedom
  - Fixed viewpoint *
    (at a given time)
  - Pan and tilt rotation
  - Zoom

Human-Computer Interface

Same type of interaction as existing non-virtual devices

- Can be used like these devices
  - Augmented reality
  - Virtual reality
- Familiar interaction
Visorama is a good idea

Less is more philosophy:
Restricts navigation to a fixed point to increase immersibility

• Avoids difficulties of virtual reality systems
• Based on panoramas (effective metaphor)

Fixed Point Restriction

Increases immersibility by simplifying interaction

• Prevents invalid operations
• No motion feedback required
• Simple and intuitive interaction
• Easy determination of focus of attention

Immersive Visualization Requirements

• Synchronization with observation device
  – Constant and interactive frame rates

• Detailed virtual environment
  – Minimum screen resolution
  – Support large panoramic images
  – Adapted image resolution

The Visorama System

• The Visorama Device
  – Executes during user interaction
  – Hardware and software components

• Authoring Environment
  • Creation of virtual world
  • Specification of user interactions

Hardware Architecture

Observation Device
Rotating Head

Software Architecture

Input Module

Output Module

Control Module

Elements for User Actions

• Viewing Parameter space (3 dimensional)
  - pan angle, tilt angle, zooming factor

• Basic Events
  - Button Pressed
  - Timer Expires

• Event Expressions
  - Complex Boolean Operations
Interaction specification

- Represented by a state diagram
  - Environment is always at a known state
  - Events cause transitions to different states
  - Actions can be executed during transitions

Example

Video Hipermídia

- Visorama Project
- Multidisciplinary Collaboration
  - VISGRAF: Technology
  - N-Imagem: Applications
- Funding
  - FINEP / Faperj / FUJB
- Main Goals
  - Real-Time Panorama Visualization
  - Applications in Historic Tourism

Rendering Panoramas in Real-Time

- Basic Requirement
  - Realism
  - Immersibility
- Mechanisms
  - Multiresolution Panoramas
  - Predictive Cache

Software Solutions

Requirements met using a combination of techniques

- Hardware implemented texture mapping
- Cache management
- Multiresolution image representation
Rendering Panoramas

• 3D Visualization

\[ R \rightarrow P = S(u,v) \]
\[ I = L(R) \]

Viewing Transformation

• Polygonal Model
• OpenGL Textures
• Virtual Camera

Multiresolution Panoramas

• Motivation
  - Arbitrary Levels of Detail (zoom)
  - Adapted Display Resolution

Image Pyramid

Adapted Multiresolution Tiling

Caches of Panorama Database

Nivel 4  Fita Magnética
Nivel 3  Servidor de Disco
Nivel 2  Disco Local
Nivel 1  RAM
Nivel 0  Placa de Video

Predictive Texture Load/Unload

• Based on Viewing Window
• Criteria
  - Temporal
  - Spatial
Adaptive Level of Detail

Example: Corcovado

Applications
- Visualization of Urban Landmarks
  - Tourism
  - Education
  - Demonstrations

Future Work
- Authoring environment
- Spatial Audio
- Animated panoramas
- Improve ergonomics

4 – Fundamentals: Concepts
- World Scenarios
- Scene Types
- Navigation
- Graphics Technologies
- Visual Output
- Interaction Types
- Medias

World Scenarios
- Wall
  - Restricted Planar 2 ½ D
- Surround
  - Unrestricted Spherical 2 ½ D
- Spatial
  - Full 3D
Scene Types

- Static
  - Discrete Change
    - Sequence of Places
- Dynamic
  - Continuous Change
    - Animated Scenes
    - Video
    - Sprites

Navigation

- Fixed Viewpoint
  - Look Around / In + Out
    - Spatial Hyperlinks
    - Temporal Transitions
    - Zoom / Traveling Motion
- Free Viewpoint
  - Fly
    - Relative 3D Motion

Visual Output

- Monoscopic
  - Fixed View
    - Planar Words
    - Surroundings Far Away
- Stereoscopic
  - Free View
    - 3D Worlds
    - Close Objects

Interaction

- Single User
  - Narrator
- Multi User
  - Avatars

Graphics Technologies

- Geometry-Based
  - 2D or 3D Models
- Image-Based
  - Images or Panoramas
- Mixed
  - Background *(images)*
  - Foreground *(models)*

Audio Technologies

- Media
  - Recorded
  - Synthesized
- Type
  - Simple
  - 3D
- Devices
  - Headphones
  - Speakers
5 – Suggestion of Projects

- Observation Deck
- Periscope
- Armored Vehicle
- Telescope
- Microscope
- Binocular
- Audio
- Games
Armored Vehicle (Visual)

Telescope

Telescope (Images)

Telescope (Annotated Images)

Microscope

Microscope (Images)
**Microscope (Simulation Images)**

**Binoculars**

**Binoculars (Video)**

**6 – Software: Device Toolkit**

- Design Goals
- Example of Application
- Basic Structure
- API

**Design Goals**

- Simple
- Easy to Use
- Work with Visorama and Mouse
  (same interface)
- C++ (Visual C)
- OpenGL
- GLUT and WINAPI

**Example of Application**

- Read and Display Device State
- Visorama Parameters:
  - Pan Angle
  - Tilt Angle
  - Zoom Button
  - Event Button
**Main Program (WINAPI)**

```c
int main (void) {
    STARTUPINFO Startup;
    GetStartupInfo(&Startup);

    BinoculoApp App(WHandle, WDef, 500, 500, "Exemplo");
    App.DisplayWindow();
    return App.MainLoop();
}
```

**Binoculo Device**

```c
class BinoculoApp : public GlWindow {
    private:
        SerialPort *SerialMonitor;
        Binoculo *EventMonitor;
        InputDevice *Device;
        BinoculoParam *param;
    public:
        BinoculoApp (HINSTANCE h, int Show, unsigned w, unsigned h, char *WinTitle);
        virtual void OpenGlRendering (HWND hwnd);
        virtual void InitAppObjects (HWND hwnd);
        virtual void InitMenu (HWND hwnd);
        virtual void Destroy (void);
        virtual void Draw (void);
        virtual void Resize (unsigned w, unsigned h);
        virtual void MouseEvents (unsigned x, unsigned y, MouseState ms);
        virtual void KeyBoardEvents (KeyBoardState kbs); 
        virtual void ControlEvents (WPARAM wParam, LPARAM lParam) {

   );
};
```

**Device Parameters**

```c
struct BinoculoParam {
    float MinPan, MaxPan; // Interval of Pan angles
    float MinTilt, MaxTilt; // Interval of Tilt angles
    float TiltAng, PanAng; // Current angles
    bool ZoomIn; // Current state of Zoom In button
    bool ZoomOut; // Current state of Zoom Out button
    bool Select; // Current state of Select Button
};
```

**Using the API**

```c
void BinoculoApp::Draw (void) {
    this->UpdateBinoculoParam();
    hdc = GetDC (hWindow);
    wglMakeCurrent (hDC, hRC);
    glClearColor (0.0,0.0,0.0,0.0); glClear (GL_COLOR_BUFFER_BIT);
    glViewport (0, 0, WinWide, WinHeight);
    // draw base
    glCircle (-0.5f, 0.5f, 0.5f, param->MinPan, param->MaxPan, 0.0f, 0.0f, 1.0f);
    ... // draw current
    glCircle (-0.5f, 0.5f, 0.5f, param->MinPan, param->PanAng, 0.0f, 1.0f, 1.0f);
    glCircle (0.5f, 0.5f, 0.5f, param->MinTilt, param->TiltAng, 1.0f, 1.0f, 0.0f);
    if (param->ZoomIn) glRect (0.25f, -0.25f, 0.5f, -0.5f, 1.0f, 0.0f, 0.0f);
    if (param->ZoomOut) glRect (0.25f, -0.6f, 0.5f, -0.85f, 1.0f, 0.0f, 0.0f);
    if (param->Select) glCircle (-0.5f, -0.5f, 0.25f, 0.0f, 360.0f, 1.0f, 0.0f, 0.0f);
    glFlush();
    SwapBuffers (hDC);
    wglMakeCurrent (NULL, NULL);
    ReleaseDC (hWindow, hDC);
}
```

**7 – Software for Panoramas**

- **Surface Type**
  - Cylindrical
  - Spherical
  - Cubical
- **Image Type**
  - Photographic
  - Synthetic
Cylindrical Panorama: *Shooting*

- Image Capture and Pre-processing

Cylindrical Panorama: *Assembly*

- Stitching the Panorama

Spherical Panorama: *Shooting*

- Fish Eye Lens:

Spherical Panorama: *Assembly*

- Convert to Polar Coordinates
- Stitch and Make Latitude / Longitude Map

Cubical Panorama

- Cube Map

Fotorama

- Java Viewer
Cuberama

- OpenGL Viewer

http://w3.impa.br/~morph/cd-morph/docs/software/cuberama.htm

7 - Links

- André Matos: Visualizador
- André Matos: Master Thesis
- Aldo Nogueira: Tech Report

8 – Ideas

- Stereoscopic Panorama
- Interactive Workbench
- Super Hi-Resolution
- Total Integration

Cache Management for Real-Time Visualization of 2D Datasets

Luiz Velho, Andre Matos, Jonas Gomes
IMPA - Instituto de Matemática Pura e Aplicada

Outline

- Tendencies and Motivation
- Principles of Cache Management
- Cache in Computer Graphics
- Algorithm for Processing of 2D Datasets
- Application: a Virtual Panorama System
- Experimental Results
- Conclusions and Current Work
Next Generation Graphics

• Tendencies
  - Fast Rendering Engines
  - High Capacity Storage Devices
  - Client-Server Architectures
• “Killer” Applications
  - Virtual Reality
  - Visual Simulators
  - CAD, GIS
  - Medical Imaging

Motivation

• Challenges
  - Very-Large Databases
  - Real-Time / Interaction
• Solutions
  - Multiresolution
  - Cache Management

Cache Management

• Resource Optimization
  - Local Memory Size: \( cs \)
  - Channel Bandwidth: \( bw \)

Cache Mechanisms

“Just-in-Time Data”

• Strategies
  - Pre-Fetch Items
  - Replace Items
• Decisions: Oracle
  Which data elements (Items)?
  * Application Dependent

Levels of Cache

• Same Problem in Cascade

Caching and Graphics

• Standard Algorithms
  - No Assumptions about Data (Sequential Fetch)
  - Exploit Usage Patterns (Least Recent Used)
• Computer Graphics Applications
  - Spatial Data Structures
  - Exploit Coherence:
    - Model (spatial)
    - Viewpoint (temporal)
  * We can do much better in Graphics...
Caching for Real-Time Graphics

- System Response Time:
  \[ \text{Frame Time} = \text{Load Objects} + \text{Render} \]
- Assumptions:
  - Objects must be in Cache for Rendering
- Real-Time and Interaction
  - Frame Rate \( \approx \) Constant
- Goal of Cache Management
  - Minimize Load Time for All Frames

Standard Cache Management

- Algorithm:
  \[
  \text{for each Frame } i \\
  \text{Load New Objects} (n_{VSi}) \\
  \text{Replace using LRU} \\
  \text{Render All Objects}
  \]
- Throughput:
  \[ \max_i(n_{VSi}) < th \]

Typical Visualization Sequence

- Drawback:
  \( n_{VSi} \) exhibits large variations from frame to frame

Using Pre-Fetching

- Strategy:
  - Select Objects and Pre-Fetch whenever Possible (i.e. when there is time left)
- Object Selection:
  1. Assume Sequence is known
  2. Use an Oracle
- Optimal Solution (case 1):
  - Maximize Response (Throughput)
  - Minimize Load Time per Frame (Latency)

Pre-Fetch Performance

- Throughput:
  \[ \max_k(\sum_{i=1}^{k} n_{VSi}) < th \]
- Latency (Accumulated):
  \[ err = \sum_{i=1}^{k} e_i \]
  with \( e_i = \begin{cases} 0 & \text{if } l_s_i \leq th \\ l_s_i - th & \text{if } l_s_i > th \end{cases} \)

Same Sequence with Pre-Fetching

- Throughput:
  \[ \sum_{i=1}^{k} n_{VSi} \]
- Latency (Accumulated):
  \[ \sum_{i=1}^{k} e_i \]
Basic Algorithm

for each Frame $i$
   Classify Objects $o_i$ using Oracle
   $c_i = $ time to render $o_i$
   Replace $th$ Objects in the Cache
   Remove $o_i$ with largest $c_i$
   Load $o_i$ with smallest $c_i$
   Render Objects

Oracle for 2D Datasets

• 2D Visualization:
  - Viewing Window (Rectangular Region)
  - Uniform Decomposition of Plane
    - Data Item is a Tile (Object)
  - Classification Criteria for Tiles:
    - Temporal
    - Spatial

Temporal Criteria

• Tile Classification:
  - Number of Frames to Enter View Window
    * Based on View Window Trajectory
    - 2nd Order Prediction (Velocity / Acceleration)

Spatial Criteria

• Tile Classification:
  - Distance from Current View Window
    * Solves Problem of Sudden Motions

Spatial Criteria

• Tile Classification:
  - Distance from Current View Window
    * Solves Problem of Sudden Motions

Application

• Visualization of Virtual Panoramas
  - Panoramic Surface
  - Viewing Projection
Processing Panoramic Images

- Tiling
- Rendering

Implementation Details

- Hardware
  - Pentium II, with 128 Mb RAM
  - Fire GL 4000, with 16 Mb Texture Memory
- Cache: level 0
  - Cache: Texture Memory
  - Storage: Main Memory
- OpenGL Rendering
  - Loading Tiles: Draw in back-buffer
  - Unloading Tiles: GL Texture Priorities

Experimental Results

- Close to Optimal

Demonstration

- Video

Conclusions

- Caching for 2D Datasets
  - Optimal Algorithm
  - Oracle for 2D Visualization
  - General Applicability (GIS, Medical, IBR)
- Extensions
  - Multiresolution
  - Multiple Levels of Cache
  - Surfaces, Volume Data