Uniquitous: Implementation and Evaluation of a Cloud-based Game System in Unity3d

IMGD M.S. Thesis Presentation
Meng Luo

Advisor: Professor Mark Claypool
Committee: Professor David Finkel
Professor Robert W. Lindeman
Background (1/2)

• What is Cloud Gaming?
  – New service based on cloud computing technology

• Why Cloud Games?
  – Convenience for players
  – Efficiency for developers
  – Reduce piracy for publishers
Background (2/2)

• Existing Cloud Gaming Systems
  ─ OnLive, Gaikai, StreamMyGame, GamingAnywhere etc.

• Cloud Games Are Growing Fast
  ─ Estimated to grow from $1 billion in 2010 to $9 billion in 2017 [1]
  ─ In 2012, Sony bought Gaikai service for $380 million and integrated the service into PlayStation in Jan. 2014 [2]
Motivation (1/2)

• Major Challenges for Cloud Gaming Providers
  – Network latency
  – Higher bandwidth required, e.g. 2 Mbps min for OnLive [3]
  – System processing delay

• Need Effective Cloud Gaming Testbed for Research and Development
  – Commercial cloud gaming systems (e.g. OnLive)
    ▪ Proprietary
  – Academic cloud gaming systems (e.g. GamingAnywhere)
    ▪ No access to and not integrated with the source code of games
Motivation (2/2)

• Uniquitous
  – More flexible and easily accessed cloud gaming system implemented with Unity3d
  – Convenient for Unity developers (1 million, 2012 to 2.5 million, 2014, 0.6 million monthly [4])
  – Allows modifications to internal structures, configurations on system parameters
  – Allows game content adjustments
    ▪ Different game scene complexities
    ▪ Different camera views

• Evaluation of Uniquitous
  – Micro evaluation
  – Macro evaluation
Outline

• Introduction
• Related Work
• Implementation
• Micro Evaluation
• Macro Evaluation
• Conclusion and Future Work
Related Work (1/2)

• Cloud Systems
  – Cloud system architecture
    ▪ Foster et al. [5] defined a four-layer model for cloud system architecture (fabric layer, Unified resource layer, Platform layer and Application layer)
  – Cloud services
    ▪ Foster et al. [5] listed the services at three different levels: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS)

• Cloud Gaming Frameworks
  – Three approaches classified by Huang et al. [6]
    ▪ Video streaming approach
    ▪ 3D graphics streaming approach
    ▪ Video streaming with post-rendering operations approach
Related Work (2/2)

• System Measurement
  – Measuring system delays
    ▪ Huang et al. [6] -- Measuring the delay of each system subcomponent of GamingAnywhere
  – Three system parameters affecting players’ experience: frame rate, game quality and game resolution
    ▪ Chang et al. [7] -- Frame rate and game quality degradation are both critical to gaming performance. Frame rate has a greater impact
    ▪ Claypool et al. [8] -- Frame rate has a greater influence on gaming performance than game resolution
Outline

- Introduction
- Related Work
- Implementation
- Micro Evaluation
- Macro Evaluation
- Conclusion and Future Work
Implementation (1/4)
Implementation (2/4)

- Image Data Flow: carry data for the game frames
  - Image Encoding: JPEG encoder
  - Image Transmission: unreliable remote procedural call (RPC)
Implementation (3/4)

- Audio data flow: carry data for the game audio
  - Audio Source: Audio listener
  - Audio Capture: `OnAudioFilterRead`, TCP socket
  - Audio Encoding & Transmission: FFMPEG
  - Audio Reception & Decoding: FFPLAY
Implementation (4/4)

• Input data flow: carry data for user input
  – Input Transmission: unreliable remote procedural call (RPC)
  – Unity Game: game scripts affected by user input
Outline

• Introduction
• Related Work
• Implementation
• Micro Evaluation
• Macro Evaluation
• Conclusion and Future Work
Micro Evaluation (1/6)

• Goal
  – Measure processing times of subcomponents of Uniquitous
  – Understand the performance bottlenecks in cloud game systems

• Experiment Setup
  – Hardware
    ▪ 12 GB RAM, Intel 3.4GHz i7-3770, AMD Radeon HD 7700 series
  – Operating System
    ▪ 64-bit Windows 7 Enterprise edition
System Parameters (2/6)

- Two Game Genres

- Eight Game Qualities

<table>
<thead>
<tr>
<th>Quality Factor (Q)</th>
<th>1</th>
<th>5</th>
<th>10</th>
<th>20</th>
<th>40</th>
<th>60</th>
<th>80</th>
<th>100</th>
</tr>
</thead>
</table>

- Nine Game Resolutions

<table>
<thead>
<tr>
<th>Game Resolution (R)</th>
<th>210 by 114</th>
<th>420 by 240</th>
<th>640 by 480</th>
<th>800 by 600</th>
<th>960 by 680</th>
<th>1280 by 720</th>
<th>1366 by 768</th>
<th>1680 by 860</th>
<th>1906 by 986</th>
</tr>
</thead>
</table>
Methodologies (3/6)

- Use Unity Pro Profiler to observe the CPU time of the component
  - Unity Project
  - Game Window

- Put time stamps in different places in the source code to measure time differences
  - Screen Capture
  - Image Encoding
  - Image Transmission

- Use Unix command “time” to get timing statistics for running the component
  - Audio Encoding & Transmission

- Experimental Results
Screen Capture (4/6)

Screen Capture Time at Nine Different Resolutions
JPEG Encoding (5/6)

- Increase game image quality increases per frame encoding time
- Increase game resolution increases per frame encoding time

Per frame encoding time versus the JPEG Quality Factor at different Resolutions (Car Tutorial)
Network Estimate (6/6)

- R: 640×480, Q: 20, AngryBots
- Uplink bitrate is fluctuating around 32 kbps
- Downlink bitrate is fluctuating around 3.5 Mbps
- Uplink traffic is much smaller than the downlink traffic
- Similar to network traffic of OnLive [9]

![Uplink network bitrate versus time](image1)

![Downlink network bitrate versus time](image2)
Outline

• Introduction
• Related Work
• Implementation
• Micro Evaluation
• **Macro Evaluation**
• Conclusion and Future Work
Macro Evaluation - Goal

• Analysis and Evaluation of performance of Uniquitous
  – Game Image Quality
  – Frame rate

• Predict Uniquitous performance under alternate configurations
Game Image Quality

• Compressed Image Samples
  – Original Image: game image from Car Tutorial
  – 200 images with 20 compression ratios and 10 resolution levels

• Objective Visual Quality Measurement
  – Peak Signal Noise Ratio (PSNR)
  – Structural Similarity Index (SSIM)

• Experiment Setup
  – Same as the Micro Evaluation
Experimental Results

SSIM values versus the JPEG quality factor among different game resolutions

- Marked increase from 1 to 15
- Modest increase from 15 to 35
- Recommended quality factor: 15 to 35
Frame Rate

- Data Samples Selection
  - Each data sample contains a different setting of JPEG encoding quality factor and resolution
  - 44 data samples for the Car Tutorial
  - 37 data samples for the AngryBots

- Frame Rate Computation
  - Use time stamps to measure frame intervals
  - Calculate the inverse of the average interval value

- Experiment Setup
  - Same as the Micro Evaluation
Experimental Results

- Increase the image quality or the resolution degrades the frame rate
- Recommended min frame rate: 15 fps [8]
- Recommended resolution: 640x480
Predicting Frame Rate (1/3)

Parallel working structure of Uniquitous Server
Predicting Frame Rate (2/3)

- Derive the Model Predicting the Frame Rate on the Server
  
  \[ F = \frac{1}{T} \]
  
  \[ T = \text{Max} (T_{1'}, T_2) + T_{\text{screenCap}} + T_{\text{transmit}} \]
  
  \[ T_{1'} = T_{\text{unity}} + T_{\text{render}} \]
  
  \[ T_2 = T_{\text{imgEn}} \]

  If \( T_2 = \text{Max} (T_{1'}, T_2) \),
  
  Then \( T = \text{Max} (T_{1'}, T_2) + T_{\text{screenCap}} + T_{\text{transmit}} + T_{\text{error}} \)
  
  \( (T_{\text{error}} \epsilon [0, 20]) \)

\( T_{1'} \): processing time of the first three components of Group 1
\( T_2 \): processing time of Group 2
\( T \): frame interval
\( F \): predicted frame rate
\( T_{\text{error}} \): error term
Predicting Frame Rate (3/3)

- Build the model to predict the client frame rate
  - Based on game Resolution (R), JPEG encoding quality factor (Q)
  - Weka Linear regression classifier (10-fold cross validation)
    - Car Tutorial: \( F_{\text{predict}} = \frac{1}{0.1348 \times R + 0.118 \times Q + 21.0} \)
    - AngryBots: \( F_{\text{predict}} = \frac{1}{0.1361 \times R + 0.1224 \times Q + 22.5} \)
  - Validation results
Validation Results

- Both models predict well
- Car: correlation coefficient is 0.995, average error percentage is 4.79%.
- Bots: correlation coefficient is 0.981, average error percentage is 9.47%.
Outline

• Introduction
• Related Work
• Implementation
• Micro Evaluation
• Macro Evaluation
• Conclusion and Future Work
Conclusions

- Uniquitous is a system for cloud game research or cloud game development.
- Uniquitous architecture: three entities and three data flows.
- The image encoding process is the processing bottleneck – processing time increases with game image quality and resolution.
- Frame rate is inversely proportional to both the game quality and the resolution.
- Recommended quality factor range for Uniquitous: 15-35, to maintain a good frame rate.
- Recommended resolution for Uniquitous: no larger than 640x480, to achieve a frame rate of 15 fps or higher.
- Models can be used by developers to choose settings for good gameplay performance.
Future Work

• Performance improvement
  – Increase the achieved frame rate
  – Support the transmission of frames of higher game quality and higher resolution

• Areas recommended for exploring with Uniquitous
  – Test with more games to include three general game genres [10]
  – Extend and deploy Uniquitous on mobile devices to evaluate its performance
Thank You!
Questions?
References


