September 15, 2017

Low-Bandwidth, Client-Based, Rendering for Gaming Videos

Ian Fischer

Shumeet Baluja

LOW-BANDWIDTH, CLIENT-BASED, RENDERING FOR GAMING VIDEOS

Gaming videos have become very popular. There is a large audience that watches other people play games, either live (e.g., online tournaments) or pre-recorded (e.g., game sessions, tournaments). Users may watch multiple videos of the same game (e.g., views of different players of the same game, different tournaments, etc.).

Gaming videos may be generated by a gaming device. A gaming device may be coupled to user input devices to receive user input. The gaming device may be, for example, a video game console, a desktop computer, a laptop, a mobile device, etc. The gaming device includes a processing device that receives the user input and generates rendering commands. The gaming device includes a graphics processing unit (GPU) that receives the rendering commands and generates the gaming video. The gaming video that is output from the GPU is recorded like any other video recorded by a camera. The GPU may format the gaming video based on the capabilities of the GPU and the video display unit that is coupled to the gaming device.

A gaming device transmits the gaming video live to a video display unit. The video display unit displays the gaming video live to a user that is providing the user input. In one implementation, the gaming device captures the gaming video, stores the gaming video (e.g., buffers the gaming video), and transmits the stored gaming video to a server device (e.g., after the recording is complete, per the available bandwidth, etc.). In another implementation, the gaming device transmits the gaming video live to the server device. The gaming device may dynamically adjust the quality of the gaming video to be transmitted to the server device based on the network connection. The server device performs similar operations (e.g., standard video compression techniques for playback, converting the gaming video into multiple settings, etc.)

on the gaming video as it does on other received videos (e.g., videos captured by a video camera or mobile device). The server device transmits the compressed gaming video to a first user device. The first user device may display the gaming video live or at a later point in time. If the gaming video is to be displayed live, the quality or lag of the gaming video may depend on the available bandwidth and processing capabilities of the first user device. The first user device displays the gaming video from the same perspective and at the same or lower quality than is displayed on the video display unit.

The transmitting of the gaming videos from the gaming device to the server device, processing (e.g., compressing) of the gaming videos by the server device, and transmitting of the gaming videos from the server device to the first user device is inefficient. The transmission of the gaming videos requires network bandwidth. The processing (e.g., compressing) of the gaming videos by the server device requires processor overhead. The perspective and quality of the gaming videos are limited to what is displayed by the video display unit of the gaming device. The first user device (e.g., a mobile device) may not be able to display the gaming videos live due to low network bandwidth or low processor capabilities.

The above and other deficiencies are addressed by providing low-bandwidth, client-based, rendering for gaming videos. A second rendering engine (e.g., rendering client) is downloaded by a second user device. The second rendering engine may have the same quality as, a lower quality than, or a higher quality than the quality of the first rendering engine. The rendering commands are transmitted via the gaming device to the server device. The server device determines the second user device is capable of generating rendered gaming video from the rendering commands. The server device transmits the rendering commands to the second user device. The rendering engine in the second user device generates the gaming video to be

displayed via the second user device.

The transmission of the rendering commands is efficient, requires less bandwidth than transmitting gaming videos, and requires less overhead for the server device than transmitting gaming videos. By receiving the rendering commands, the second user device can display a perspective (e.g., camera position, viewpoint of other players, any gaming video seen from the game) that is different than that of the gaming video displayed via the video display unit. For example, the second user device may display video feeds of different players, video feeds of different tournaments, camera angles that are different from the camera angle the video display unit is displaying, etc. The second user device can pause the rendered video game and change camera position. The rendered gaming video displayed by the second user device is much more interactive than the compressed gaming video displayed by the first user device. The second user device can display a quality of rendered gaming video that is different than that of the gaming video displayed via the video display unit. The second user device can receive the rendering commands over a low-bandwidth connection, generate the gaming videos, and display the gaming videos live. Receiving rendering commands yields a less jittery and an uncompressed viewing experience for the user of the second device than receiving gaming videos.

FIG. 1 illustrates an example system architecture. The system architecture includes a gaming device, a user input device, a video display unit, a server device, a first user device, and a second user device. The system architecture in FIG. 1 is described in relation to a gaming device and gaming video, however the system architecture in FIG. 1 can be applied to other computing systems and other multimedia items that are based on rendering commands, graphics library commands, user sequence of inputs, etc.

The gaming device may be a gaming console, computing device, smartphone, tablet,

laptop, desktop, etc. The gaming device includes a processing device and a GPU. The GPU includes a first rendering engine. The gaming device is coupled to a user input device, a video display unit, and a server device. The user input device may include an alpha-numeric input device (e.g., keyboard), a cursor control device (e.g., mouse), touchpad, buttons, video game controller, microphone, video camera, motion sensors, etc.

The gaming device receives user input via the user input device. The processing device processes the user input to generate rendering commands (e.g., commands to the renderer, rendering calls). The processing device may execute a gaming program (e.g., gaming engine) to generate the rendering commands. The processing device may generate the rendering commands via an input processing unit, game logic, game artificial intelligence, identifying game assets, etc. The first rendering engine of the GPU receives the rendering commands and generates a gaming video. The first rendering engine may determine a video resolution for the gaming video, determine whether the gaming video is three-dimensional (3D) or two-dimensional (2D), obtain game assets for the gaming video, determine shading for the gaming video, determine lighting for the gaming video, etc. The first rendering engine may generate the gaming video based on the GPU processing capabilities, the gaming device processing capabilities, the video display unit capabilities (e.g., resolution), etc. The video display unit (e.g., television, monitor, etc.) receives the gaming video and displays the gaming video to the user that is providing the user input via the user input device.

The gaming device transmits the gaming video and the rendering commands to the server device. The rendering commands are significantly smaller (e.g., 1% or less) than the size of the gaming videos. In one implementation, the rendering commands are transmitted as a separate stream from the gaming video. In another implementation, the rendering commands are

transmitted with the gaming video (e.g., as the equivalent of "comments" in the gaming video).

The server device may process (e.g., compress, produce different formats) the gaming video to generated compressed gaming video. The server device may determine that the first user device does not include a rendering engine that is compatible with the rendering commands. The server device transmits the compressed gaming video to the first user device.

The server device determines that the second user device includes a second rendering engine that is compatible with the rendering commands. For example, the server device may determine the second rendering engine is installed on the second user device. In another example, the server device may determine the gaming program (e.g., gaming engine) is installed on the second user device. In another example, the server device may determine a browser plugin that is compatible with the rendering commands and that is installed on the second user device. The server device transmits the rendering commands to the second user device. The second user device generates rendered gaming video (e.g., via the second rendering engine or the game engine running in the background) and displays the rendered gaming video. The second user device displays the rendered gaming video via a user interface associated with the server device. The second user device may execute an end-user application that is associated with the server device to display the rendered gaming video.

The gaming device may download and execute a gaming program to generate the rendering commands and the gaming video. The gaming program may be a game engine that includes a rendering engine. Different gaming programs may share the same rendering engine. Each gaming program may have its own game assets (e.g., game art, game artifacts, characters, objects, three-dimensional shapes, textures, models, etc.).

In one implementation, the second rendering engine is a game-specific rendering engine

that matches the gaming program executed on the gaming device. In another implementation, the second rendering engine is from a different gaming program than the gaming program executed on the gaming device and the second rendering engine has downloaded the game assets to be compatible with the rendering commands. The different gaming program executed on the second user device may include the same type of rendering engine as the rendering engine of the gaming program executed on the gaming device.

Downloading the game-specific rendering engine or a different rendering engine with the game assets provides a better and more efficient gaming video experience. The game-specific rendering engine, the different rendering engine, or the game assets may be downloaded as an Internet browser plugin. The game-specific rendering engine, the different rendering engine, or the game assets may be downloaded while the second user device has a higher bandwidth connection and not a lower bandwidth connection (e.g., mobile connection).

In one implementation, the first rendering engine and the second rendering engine may produce gaming videos of different qualities. For example, mobile devices (e.g., cell phone, tablet) may have a rendering engine that is scaled back (e.g., a lower quality) from the rendering engine of a gaming console to meet specific processor capabilities. The first rendering engine may take into account the capabilities of the gaming device and the video display unit. The second rendering engine may take into account the capabilities of the second user device and any external video display units used to display the gaming video. In one implementation, the video display unit of the gaming device has a lower resolution (e.g., 1080p, High Definition (HD)) than the resolution of video display units of first user device and second user device (e.g., 4K, Ultra High Definition (UHD)). The first user device will receive the gaming video compressed to display at the lower resolution (e.g., HD), whereas the second user device will receive the

rendering commands and generate the gaming video uncompressed to display at the higher resolution (e.g., 4K).

The gaming video can be displayed on the second user device in different formats than the format that is displayed by the video display unit of the gaming device. For example, the second user device can generate (e.g., on-the-fly) three-dimensional (3D), virtual reality, cardboard style, etc. experiences out of the gameplay. Full 3D may be reconstructed by the second rendering engine even if the original player was not playing in 3D mode.

Many gaming programs have different detail-level modes in which the gaming programs can run. This is useful for adapting the capabilities of different computers (e.g., second user device, gaming device) on which the gaming programs are run. For example, for a gaming program running on a weak processing device, details, objects, and lighting effects can be removed and/or reduced. Each rendering command may be tagged with an identifier that indicates a detail-level mode (e.g., indicates what level of detail the gaming video generated from the rendering command is to have). In one implementation, the server device determines the second user device has low bandwidth and a low-capacity processing device and only transmits rendering commands that would be executed at the lowest detail level. In another implementation, the server device transmits rendering commands at different detail levels to the second user device and the second user device selects (e.g., filters) the rendering command that matches the capacity of the second user device.

The voice of the user of the gaming device (e.g., the game player) may be recorded to get reactions and play-by-play commentary as the game is being played. This audio can be transmitted as an auxiliary audio stream that is coordinated through audio stream timestamps to the timestamps in the rendering commands.

In one implementation, the server device receives the rendering commands and the rendering is performed by the server device or a device coupled to the server device to generate rendered gaming video (e.g., the server device or a device coupled to the server device includes a third rendering engine). The server device may deliver or store the rendered gaming video subsequent to the rendering based on the rendering commands of the user of the gaming device. The gaming device could transmit rendering commands live (e.g., stream in real time) despite having a poor upload connection or poor speed, the server device could generate the rendered gaming video, and the server device could transmit the rendered gaming video live to a user device.

In one implementation, the gaming device transmits rendering commands to the server device, the server device transmits the rendering commands to the second user device, and the second rendering engine of the second user device generates the rendered gaming video.

In another implementation, the gaming device transmits graphics library commands to the server device, the server device transmits the graphics library commands to a user device, and the second rendering engine of the user device accesses the graphics library downloaded on the user device to generate the rendered gaming video. The graphics library commands may be low-level graphics library commands generated by the GPU or the processing device. The graphics library downloaded on the user device may be a compatible low-level graphics library. The rendered gaming video may have identical content as the gaming video shown via the video display unit of the gaming device.

The user device may execute an end-user application to display the rendered gaming video. In one implementation, the end-user application may provide a compatibility layer for the different graphics libraries that the end-user application supports. In another implementation, the

end-user application requests the graphics library commands instead of the video stream in response to detecting the user device supports the graphics library used by the gaming device. A first type of gaming device (e.g., a gaming console) may have a different type of graphics library than a second type of user device (e.g., a laptop, a desktop computer). The user device or end-user application may determine whether the user device has downloaded a graphics library of the type of graphics library used by the gaming device that is transmitting the graphics library commands. In one implementation, to generate the gaming video from the graphics library commands, the user device downloads the game assets used by the stream of graphics library commands (e.g., the user device does not download the entire graphics library used by the gaming program).

In another implementation, the gaming device transmits the user input (e.g., user sequence of inputs) and the game state to the server device. The server device determines the user device is capable of generating gaming video from the user input and game state. The server device may determine the second user device has downloaded the game engine. The server device may determine the second user device has downloaded the gaming program. The gaming program may be a deterministic program that generates the same output from the same user input and game state (e.g., not a multi-player game, etc.). The server device transmits the user input and game state to a user device and the user device (e.g., gaming engine of the user device) generates the rendered gaming video. In another implementation, the gaming device may transmit a different type of commands or input that a user device is capable of using to generate rendered gaming video.

The implementations described herein may not be specific to a gaming device and gaming videos. For example, a computing device may transmit commands (e.g., rendering

commands, graphics library commands) or user input to a server device. The computing device may generate a multimedia item from the commands or input (e.g., via a GPU, processing device, etc.). The server device may determine a user device is capable of generating the multimedia item and may transmit the commands or input to the user device.

ABSTRACT

A system for low-bandwidth, client-based, rendering for gaming videos is described. The system may include a gaming device, server device, and user devices. The gaming device may include a processing device and graphics processing unit (GPU). The processing device receives user input and generates rendering commands from the user input. A first rendering unit of the GPU generates gaming video from the rendering commands. The server device receives the gaming video and the rendering commands from the gaming device. The server device determines the first user device is not compatible with the rendering commands, compresses the gaming video, and transmits the compressed gaming video to the first user device. The server device determines the second user device is compatible with the rendering commands and transmits the rendering commands to the second user device. The second rendering engine of the second user device generates rendered gaming video from the rendering commands.

Keywords: copy, render, engine, content, audio, command, virtual reality, library, low, client device, video game, server, timestamp

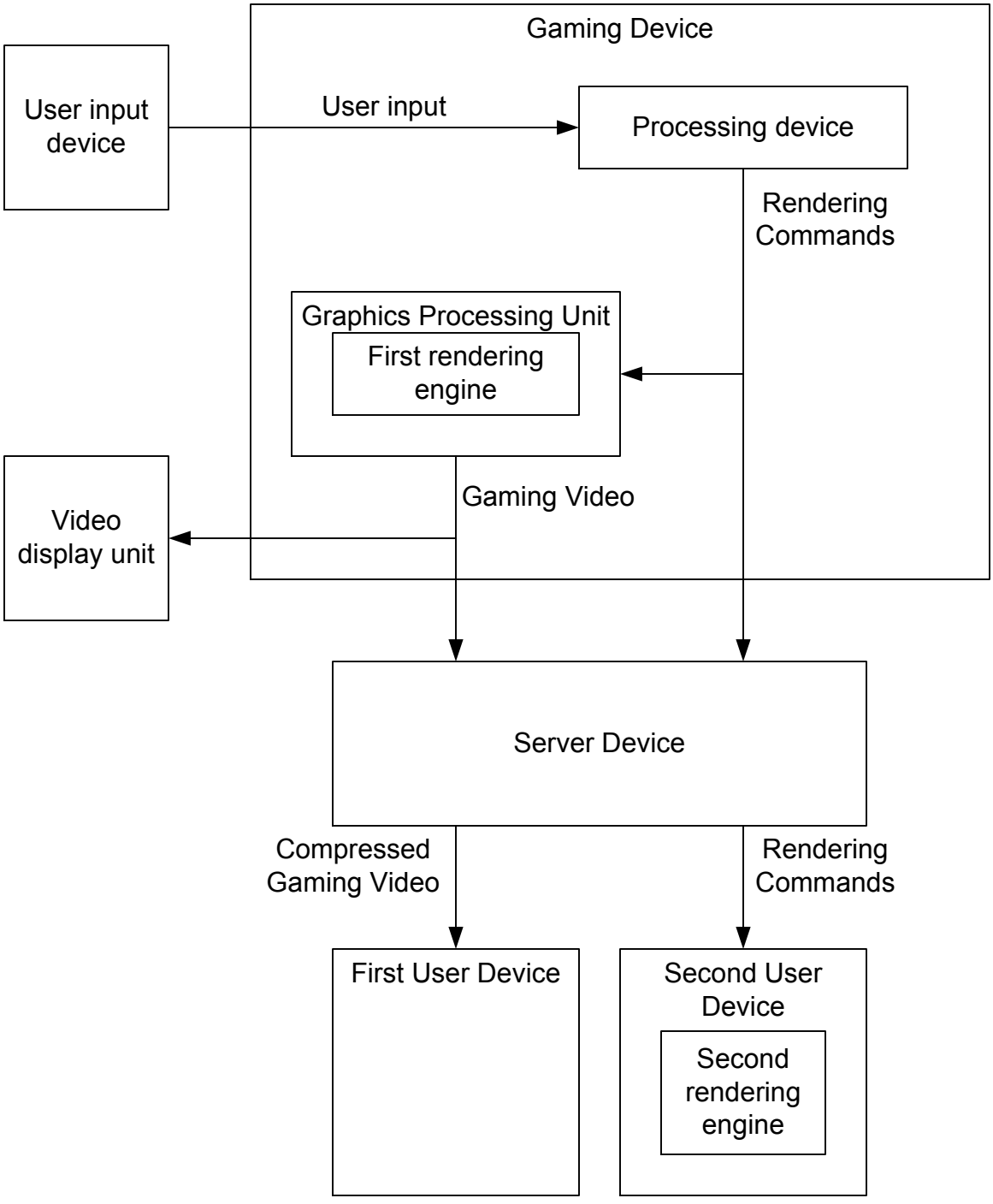
Gaming Device
User input device
User input
Processing device
Rendering Commands
Graphics Processing Unit
First rendering engine
Video display unit
Gaming Video
Server Device
Compressed Gaming Video
Rendering Commands
First User Device
Second User Device
Second rendering engine

FIG. 1